



Guia do Desenvolvedor

Amazon Elastic Compute Cloud



Amazon Elastic Compute Cloud: Guia do Desenvolvedor

Copyright © 2024 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

As marcas comerciais e imagens de marcas da Amazon não podem ser usadas no contexto de nenhum produto ou serviço que não seja da Amazon, nem de qualquer maneira que possa gerar confusão entre os clientes ou que deprecie ou desprestige a Amazon. Todas as outras marcas comerciais que não pertencem à Amazon pertencem a seus respectivos proprietários, que podem ou não ser afiliados, patrocinados pela Amazon ou ter conexão com ela.

Table of Contents

Acesso programático ao Amazon EC2	1
Service endpoints	1
IPv4endpoints	2
Endpoints de pilha dupla (IPv4e) IPv6	2
Endpoints de serviço por região	3
Especificar endpoints	7
Consistência eventual	9
Potência igual	11
Idempotência na Amazon EC2	12
RunInstances idempotência	15
Exemplos	16
Repita as recomendações para solicitações idempotentes	18
APIlimitação de solicitações	19
Como o controle de utilização é aplicado	20
Limites de controle de utilização	21
Limitação do API monitor	35
Tentativas repetidas e recuo exponencial	35
Solicitar um aumento de limite de	36
Usando o AWS CLI	38
Saiba mais sobre o AWS CLI	38
Usando AWS CloudFormation	39
Amazon EC2 e modelos AWS CloudFormation	39
Recursos para o Amazon EC2	39
Saiba mais sobre AWS CloudFormation	43
Usando um AWS SDK	44
Exemplos de código para a Amazon EC2 API	44
Saiba mais sobre o AWS SDKs	44
API de baixo nível para Amazon EC2	45
Console-to-Code	46
Exemplos de código	47
Conceitos básicos	67
Olá Amazon EC2	74
Aprender os conceitos básicos	87
Ações	304

Cenários	1061
Criar e gerenciar um serviço resiliente	1062
Monitore API solicitações usando CloudWatch	1231
Ative as EC2 API métricas da Amazon	1231
EC2APIMétricas e dimensões da Amazon	1232
Metrics	1232
Dimensões	1233
Retenção de dados métricos	1233
Monitoramento de solicitações feitas em seu nome	1234
Faturamento	1234
Trabalhando com a Amazon CloudWatch	1234
Visualizando CloudWatch métricas	1234
Criação de CloudWatch alarmes	1235
.....	mccxxxviii

Acesso programático ao Amazon EC2

Você pode criar e gerenciar seus recursos do Amazon EC2 usando AWS Management Console ou uma interface programática. Para obter informações sobre o uso do console do Amazon EC2, consulte o Guia do usuário do [Amazon EC2](#).

Como funciona

- [Endpoints do Amazon EC2](#)
- [Consistência eventual](#)
- [Idempotência](#)
- [Limitação de solicitações](#)

Interfaces programáticas

- [AWS Command Line Interface \(AWS CLI\)](#)
- [AWS CloudFormation](#)
- [AWS SDKs](#)
- [API de baixo nível](#)

Conceitos básicos

- [Exemplos de código](#)
- [Console-to-Code](#)

Monitorar

- [AWS CloudTrail](#)
- [Monitore solicitações](#)

Endpoints EC2 de serviço da Amazon

Um endpoint é URL aquele que serve como ponto de entrada para um serviço AWS web. A Amazon EC2 oferece suporte aos seguintes tipos de endpoints:

- [IPv4endpoints](#)
- [Endpoints de pilha dupla \(suportam tanto quanto\) IPv4 IPv6](#)
- [FIPSendpoints](#)

Ao fazer uma solicitação, você pode especificar o endpoint a ser usado. Se você não especificar um endpoint, o IPv4 endpoint será usado por padrão. Para usar outro tipo de endpoint, você deve especificá-lo em sua solicitação. Para obter exemplos de como fazer isso, consulte [Especificar endpoints](#). Para obter uma tabela dos endpoints disponíveis, consulte [Endpoints de serviço por região](#).

IPv4endpoints

IPv4os endpoints oferecem suporte somente ao IPv4 tráfego. IPv4os endpoints estão disponíveis para todas as regiões.

Se você especificar o endpoint geral, `ec2.amazonaws.com`, usaremos o endpoint para `us-east-1`. Para usar uma região diferente, especifique o endpoint associado a ela. Por exemplo, se você especificar `ec2.us-east-2.amazonaws.com` como endpoint, direcionaremos sua solicitação para o endpoint `us-east-2`.

IPv4os nomes de endpoints usam a seguinte convenção de nomenclatura:

- `service.region.amazonaws.com`

Por exemplo, o nome do IPv4 endpoint para a `eu-west-1` região é `ec2.eu-west-1.amazonaws.com`.

Endpoints de pilha dupla (IPv4e) IPv6

Os endpoints de pilha dupla oferecem suporte tanto ao tráfego quanto ao tráfego. IPv4 IPv6 Quando você faz uma solicitação para um endpoint de pilha dupla, o endpoint é URL resolvido para um IPv4 endereço IPv6 ou, dependendo do protocolo usado pela rede e pelo cliente.

A Amazon EC2 oferece suporte somente a endpoints regionais de pilha dupla, o que significa que você deve especificar a região como parte do nome do endpoint. Os nomes de endpoints de pilha dupla usam a seguinte convenção de nomenclatura:

- `ec2.region.api.aws`

Por exemplo, o nome do endpoint de pilha dupla para a região eu-west-1 é `ec2.eu-west-1.api.aws`.

Endpoints de serviço por região

A seguir estão os endpoints de serviço da AmazonEC2. Para obter mais informações sobre regiões, consulte [Regiões e zonas de disponibilidade](#) no Guia EC2 do usuário da Amazon.

Nome da região	Região	Endpoint	Protocolo
Leste dos EUA (Ohio)	us-east-2	<code>ec2.us-east-2.amazonaws.com</code>	HTTPe
		<code>ec2-fips.us-east-2.amazonaws.com</code>	HTTPS
		<code>ec2.us-east-2.api.aws</code>	HTTPS
Leste dos EUA (Norte da Virgínia)	us-east-1	<code>ec2.us-east-1.amazonaws.com</code>	HTTPe
		<code>ec2-fips.us-east-1.amazonaws.com</code>	HTTPS
		<code>ec2.us-east-1.api.aws</code>	HTTPS
Oeste dos EUA (N. da Califórnia)	us-west-1	<code>ec2.us-west-1.amazonaws.com</code>	HTTPe
		<code>ec2-fips.us-west-1.amazonaws.com</code>	HTTPS
		<code>ec2.us-west-1.api.aws</code>	HTTPS
Oeste dos EUA (Oregon)	us-west-2	<code>ec2.us-west-2.amazonaws.com</code>	HTTPe
		<code>ec2-fips.us-west-2.amazonaws.com</code>	HTTPS
		<code>ec2.us-west-2.api.aws</code>	HTTPS

Nome da região	Região	Endpoint	Protocolo
África (Cidade do Cabo)	af-south-1	ec2.af-south-1.amazonaws.com	HTTPe
		ec2.af-south-1.api.aws	HTTPS
			HTTPS
Ásia-Pacífico (Hong Kong)	ap-east-1	ec2.ap-east-1.amazonaws.com	HTTPe
		ec2.ap-east-1.api.aws	HTTPS
			HTTPS
Ásia-Pacífico (Hyderabad)	ap-south-2	ec2.ap-south-2.amazonaws.com	HTTPS
Ásia-Pacífico (Jacarta)	ap-southeast-3	ec2.ap-southeast-3.amazonaws.com	HTTPS
Ásia-Pacífico (Malásia)	ap-southeast-5	ec2.ap-southeast-5.amazonaws.com	HTTPS
Ásia-Pacífico (Melbourne)	ap-southeast-4	ec2.ap-southeast-4.amazonaws.com	HTTPS
Ásia-Pacífico (Mumbai)	ap-south-1	ec2.ap-south-1.amazonaws.com	HTTPe
		ec2.ap-south-1.api.aws	HTTPS
			HTTPS
Ásia-Pacífico (Osaka)	ap-northeast-3	ec2.ap-northeast-3.amazonaws.com	HTTPe HTTPS

Nome da região	Região	Endpoint	Protocolo
Ásia-Pacífico (Seul)	ap-northeast-2	ec2.ap-northeast-2.amazonaws.com	HTTPe
		ec2.ap-northeast-2.api.aws	HTTPS
			HTTPS
Ásia-Pacífico (Singapura)	ap-southeast-1	ec2.ap-southeast-1.amazonaws.com	HTTPe
		ec2.ap-southeast-1.api.aws	HTTPS
			HTTPS
Ásia-Pacífico (Sydney)	ap-southeast-2	ec2.ap-southeast-2.amazonaws.com	HTTPe
		ec2.ap-southeast-2.api.aws	HTTPS
			HTTPS
Ásia-Pacífico (Tóquio)	ap-northeast-1	ec2.ap-northeast-1.amazonaws.com	HTTPe
		ec2.ap-northeast-1.api.aws	HTTPS
			HTTPS
Canadá (Central)	ca-central-1	ec2.ca-central-1.amazonaws.com	HTTPe
		ec2-fips.ca-central-1.amazonaws.com	HTTPS
		ec2.ca-central-1.api.aws	HTTPS
			HTTPS
Oeste do Canadá (Calgary)	ca-west-1	ec2.ca-west-1.amazonaws.com	HTTPS
		ec2-fips.ca-west-1.amazonaws.com	HTTPS
Europa (Frankfurt)	eu-central-1	ec2.eu-central-1.amazonaws.com	HTTPe
		ec2.eu-central-1.api.aws	HTTPS
			HTTPS

Nome da região	Região	Endpoint	Protocolo
Europa (Irlanda)	eu-west-1	ec2.eu-west-1.amazonaws.com	HTTPe
		ec2.eu-west-1.api.aws	HTTPS
			HTTPS
Europa (Londres)	eu-west-2	ec2.eu-west-2.amazonaws.com	HTTPe
		ec2.eu-west-2.api.aws	HTTPS
			HTTPS
Europa (Milão)	eu-south-1	ec2.eu-south-1.amazonaws.com	HTTPe
		ec2.eu-south-1.api.aws	HTTPS
			HTTPS
Europa (Paris)	eu-west-3	ec2.eu-west-3.amazonaws.com	HTTPe
		ec2.eu-west-3.api.aws	HTTPS
			HTTPS
Europa (Espanha)	eu-south-2	ec2.eu-south-2.amazonaws.com	HTTPS
Europa (Estocolmo)	eu-north-1	ec2.eu-north-1.amazonaws.com	HTTPe
		ec2.eu-north-1.api.aws	HTTPS
			HTTPS
Europa (Zurique)	eu-central-2	ec2.eu-central-2.amazonaws.com	HTTPS
Israel (Tel Aviv)	il-central-1	ec2.il-central-1.amazonaws.com	HTTPS
Oriente Médio (Barém)	me-south-1	ec2.me-south-1.amazonaws.com	HTTPe
		ec2.me-south-1.api.aws	HTTPS
			HTTPS

Nome da região	Região	Endpoint	Protocolo
Oriente Médio (UAE)	me-central-1	ec2.me-central-1.amazonaws.com	HTTPS
América do Sul (São Paulo)	sa-east-1	ec2.sa-east-1.amazonaws.com	HTTPe
		ec2.sa-east-1.api.aws	HTTPS
			HTTPS
AWS GovCloud (Leste dos EUA)	us-gov-east-1	ec2.us-gov-east-1.amazonaws.com	HTTPS
		ec2.us-gov-east-1.api.aws	HTTPS
AWS GovCloud (Oeste dos EUA)	us-gov-west-1	ec2.us-gov-west-1.amazonaws.com	HTTPS
		ec2.us-gov-west-1.api.aws	HTTPS

Especificar endpoints

Esta seção fornece alguns exemplos de como especificar um endpoint ao fazer uma solicitação.

AWS CLI

Os exemplos a seguir mostram como especificar um endpoint para a `us-east-2` região usando o AWS CLI

- Pilha dupla

```
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

Os exemplos a seguir mostram como especificar um endpoint para a us-east-2 região usando o AWS SDK for Java 2.x

- Pilha dupla

```
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

Os exemplos a seguir mostram como especificar um endpoint para a eu-west-1 região usando o AWS SDK for Java 1.x.

- Pilha dupla

```
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();
```

```
"https://ec2.eu-west-1.amazonaws.com",  
"eu-west-1"))  
.build();
```

AWS SDK for Go

Os exemplos a seguir mostram como especificar um endpoint para a us-east-1 região usando o AWS SDK for Go

- Pilha dupla

```
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")  
})
```

Consistência eventual na Amazon EC2 API

A Amazon EC2 API segue um modelo de consistência eventual, devido à natureza distribuída do sistema que suporta o API. Isso significa que o resultado de um API comando executado que afeta seus EC2 recursos da Amazon pode não ser imediatamente visível para todos os comandos subsequentes que você executar. Você deve ter isso em mente ao executar um API comando que segue imediatamente um API comando anterior.

A consistência eventual pode afetar a maneira como você gerencia seus recursos. Por exemplo, se você executar um comando para criar um recurso, ele eventualmente ficará visível para outros comandos. Isso significa que, se você executar um comando para modificar ou descrever o recurso que acabou de criar, seu ID pode não ter se propagado por todo o sistema e você receberá um erro respondendo que o recurso não existe.

Para gerenciar a consistência eventual, é possível fazer o seguinte:

- Confirme o estado do recurso antes de executar um comando para modificá-lo. Execute o comando `Describe` apropriado usando um algoritmo de recuo exponencial para garantir tempo suficiente para que o comando anterior se propague pelo sistema. Para fazer isso, execute o comando `Describe` repetidamente, começando com alguns segundos de espera e aumentando gradualmente até cinco minutos de espera.
- Adicione o tempo de espera entre os comandos subsequentes, mesmo que um comando `Describe` exiba uma resposta precisa. Aplique um algoritmo de recuo exponencial começando com alguns segundos de espera e aumente gradualmente até cerca de cinco minutos de espera.

Exemplos de eventuais erros de consistência

A seguir estão exemplos de códigos de erro que você pode encontrar como resultado de uma eventual consistência.

- `InvalidInstanceID.NotFound`

Se você executar o `RunInstances` comando com êxito e, em seguida, executar imediatamente outro comando usando o ID da instância fornecido na resposta de `RunInstances`, ele poderá retornar um `InvalidInstanceID.NotFound` erro. Isso não significa que a instância não exista.

Alguns comandos específicos que podem ser afetados são:

- `DescribeInstances`: para confirmar o estado real da instância, execute esse comando usando um algoritmo de recuo exponencial.
- `TerminateInstances`: para confirmar o estado da instância, primeiro execute o `DescribeInstances` comando usando um algoritmo de recuo exponencial.

Important

Se você receber um `InvalidInstanceID.NotFound` erro após a execução de `TerminateInstances`, isso não significa que a instância foi ou será encerrada. Sua instância ainda pode estar em execução. É por isso que é importante primeiro confirmar o estado da instância usando `DescribeInstances`.

- `InvalidGroup.NotFound`

Se você executar o `CreateSecurityGroup` comando com êxito e, em seguida, executar imediatamente outro comando usando o ID do grupo de segurança fornecido na resposta de `CreateSecurityGroup`, ele poderá retornar um `InvalidGroup.NotFound` erro. Para confirmar o estado do grupo de segurança, execute o `DescribeSecurityGroups` comando usando um algoritmo de recuo exponencial.

- `InstanceLimitExceeded`

Você solicitou mais instâncias do que o limite atual permite para o tipo de instância especificado. Você pode atingir esse limite inesperadamente se estiver iniciando e encerrando instâncias rapidamente, pois as instâncias encerradas contam para seu limite de instâncias por um tempo depois de serem encerradas.

Garantindo a idempotência nas solicitações da Amazon EC2 API

Quando você faz uma solicitação de mutação, a API solicitação normalmente retorna um resultado antes que os fluxos de trabalho assíncronos da operação sejam concluídos. As operações também podem expirar ou encontrar outros problemas no servidor antes de serem concluídas, mesmo que a solicitação já tenha retornado um resultado. Isso pode dificultar na hora de determinar se a solicitação foi bem-sucedida ou não, e pode levar a várias novas tentativas para garantir que a operação seja concluída com êxito. No entanto, se a solicitação original e as tentativas subsequentes forem bem-sucedidas, a operação será concluída várias vezes. Isso significa que você pode criar mais recursos do que pretendia.

A idempotência garante que uma API solicitação seja concluída no máximo uma vez. Com uma solicitação idempotente, se a solicitação original for concluída com êxito, todas as novas tentativas subsequentes serão concluídas com êxito sem realizar nenhuma ação. No entanto, o resultado pode conter informações atualizadas, como o status atual da criação.

Conteúdo

- [Idempotência na Amazon EC2](#)
- [RunInstances idempotência](#)
- [Exemplos](#)
- [Repita as recomendações para solicitações idempotentes](#)

Idempotência na Amazon EC2

As API ações a seguir são idempotentes por padrão e não exigem configuração adicional. Os AWS CLI comandos correspondentes também oferecem suporte à idempotência por padrão.

Idempotente por padrão

- AssociateAddress
- CreateVpnConnection
- DisassociateAddress
- ReplaceNetworkAclAssociation
- TerminateInstances

As API ações a seguir oferecem suporte opcional à idempotência usando um token de cliente. Os AWS CLI comandos correspondentes também oferecem suporte à idempotência usando um token de cliente. Um token de cliente é uma sequência exclusiva, com distinção entre maiúsculas e minúsculas, de até 64 ASCII caracteres. Para fazer uma API solicitação idempotente usando uma dessas ações, especifique um token de cliente na solicitação. Você não deve reutilizar o mesmo token de cliente para outras API solicitações. Se você tentar novamente uma solicitação concluída com êxito usando o mesmo token de cliente e os mesmos parâmetros, a nova tentativa será bem-sucedida sem realizar nenhuma ação adicional. Se você tentar novamente uma solicitação bem-sucedida usando o mesmo token de cliente, mas um ou mais dos parâmetros forem diferentes, exceto a região ou a zona de disponibilidade, a nova tentativa falhará com um `IdempotentParameterMismatch` erro.

Idempotente usando um token de cliente

- AllocateHosts
- AllocateIamPoolCidr
- AssociateClientVpnTargetNetwork
- AssociateIamResourceDiscovery
- AttachVerifiedAccessTrustProvider
- AuthorizeClientVpnIngress
- CopyFpgaImage
- CopyImage

- CreateCapacityReservation
- CreateCapacityReservationFleet
- CreateClientVpnEndpoint
- CreateClientVpnRoute
- CreateEgressOnlyInternetGateway
- CreateFleet
- CreateFlowLogs
- CreateFpgaImage
- CreateInstanceConnectEndpoint
- CreateIam
- CreateIamPool
- CreateIamResourceDiscovery
- CreateIamScope
- 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`

Tipos de idempotência

- **Regional** — As solicitações são idempotentes em cada região. No entanto, você pode usar a mesma solicitação, incluindo o mesmo token de cliente, em uma região diferente.
- **Zonal** — As solicitações são idempotentes em cada zona de disponibilidade em uma região. Por exemplo, se você especificar o mesmo token de cliente em duas chamadas para `AllocateHosts` na mesma região, as chamadas serão bem-sucedidas se especificarem valores diferentes para o `AvailabilityZone` parâmetro.

RunInstances idempotência

A [RunInstances](#) API usa idempotência regional e zonal.

O tipo de idempotência usada depende de como você especifica a zona de disponibilidade em sua solicitação. RunInstances API A solicitação usa idempotência zonal nos seguintes casos:

- Se você especificar explicitamente uma zona de disponibilidade usando o `AvailabilityZone` parâmetro no tipo de dados de posicionamento
- Se você especificar implicitamente uma zona de disponibilidade usando o parâmetro `SubnetId`

Se você não especificar explícita ou implicitamente uma zona de disponibilidade, a solicitação usará a idempotência regional.

Idempotência zonal

A idempotência zonal garante que uma RunInstances API solicitação seja idempotente em cada zona de disponibilidade em uma região. Isso garante que uma solicitação com o mesmo token de cliente possa ser concluída somente uma vez em cada zona de disponibilidade em uma região. No entanto, o mesmo token de cliente pode ser usado para iniciar instâncias em outras zonas de disponibilidade na região.

Por exemplo, se você enviar uma solicitação idempotente para iniciar uma instância na Zona de `us-east-1a` Disponibilidade e depois usar o mesmo token de cliente em uma solicitação na Zona de `us-east-1b` Disponibilidade, lançaremos instâncias em cada uma dessas Zonas de Disponibilidade. Se um ou mais dos parâmetros forem diferentes, as tentativas subsequentes com o

mesmo token de cliente nessas zonas de disponibilidade retornarão com êxito sem realizar nenhuma ação adicional ou falharão com um `IdempotentParameterMismatch` erro.

Idempotência regional

A idempotência regional garante que uma `RunInstances` API solicitação seja idempotente em uma região. Isso garante que uma solicitação com o mesmo token de cliente possa ser concluída somente uma vez dentro de uma região. No entanto, exatamente a mesma solicitação, com o mesmo token de cliente, pode ser usada para iniciar instâncias em uma região diferente.

Por exemplo, se você enviar uma solicitação idempotente para iniciar uma instância na `us-east-1` região e depois usar o mesmo token de cliente em uma solicitação na `eu-west-1` região, lançaremos instâncias em cada uma dessas regiões. Se um ou mais dos parâmetros forem diferentes, as tentativas subsequentes com o mesmo token de cliente nessas regiões retornarão com êxito sem realizar nenhuma ação adicional ou falharão com um `IdempotentParameterMismatch` erro.

Tip

Se uma das zonas de disponibilidade na região solicitada não estiver disponível, as `RunInstances` solicitações que usam a idempotência regional podem falhar. Para aproveitar os recursos da zona de disponibilidade oferecidos pela AWS infraestrutura, recomendamos que você use a idempotência zonal ao iniciar instâncias. `RunInstances` solicitações que usam idempotência zonal e têm como alvo uma zona de disponibilidade disponível são bem-sucedidas mesmo que outra zona de disponibilidade na região solicitada não esteja disponível.

Exemplos

AWS CLI exemplos de comando

Para tornar um AWS CLI comando idempotente, adicione a opção. `--client-token`

Exemplo 1: Idempotência

O comando [allocate-hosts](#) a seguir usa idempotência, pois inclui um token de cliente.

```
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
```

Exemplo 2: idempotência regional de instâncias de execução

O comando [run-instances](#) a seguir usa idempotência regional, pois inclui um token de cliente, mas não especifica explícita ou implicitamente uma zona de disponibilidade.

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

Exemplo 3: idempotência zonal de instâncias de execução

O comando [run-instances](#) a seguir usa idempotência zonal, pois inclui um token de cliente e uma zona de disponibilidade especificada explicitamente.

```
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 exemplos de solicitação

Para tornar uma API solicitação idempotente, adicione o parâmetro. ClientToken

Exemplo 1: Idempotência

A [AllocateHosts](#) API solicitação a seguir usa idempotência, pois inclui um token de cliente.

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

Exemplo 2: RunInstances idempotência regional

A [RunInstances](#) API solicitação a seguir usa idempotência regional, pois inclui um token de cliente, mas não especifica explícita ou implicitamente uma zona de disponibilidade.

```
https://ec2.amazonaws.com/?Action=RunInstances  
&ImageId=ami-3ac33653  
&MaxCount=1
```

```
&MinCount=1
&KeyName=my-key-pair
&ClientToken=550e8400-e29b-41d4-a716-446655440000
&AUTHPARAMS
```

Exemplo 3: RunInstances idempotência zonal

A [RunInstances](#) API solicitação a seguir usa idempotência zonal, pois inclui um token de cliente e uma zona de disponibilidade especificada explicitamente.

```
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
```

Repita as recomendações para solicitações idempotentes

A tabela a seguir mostra algumas respostas comuns que você pode obter para API solicitações idempotentes e fornece recomendações de novas tentativas.

Resposta	Recomendação	Comentários
200 (OK)	Não repetir	A solicitação original foi concluída com êxito. Qualquer repetição subsequente é retornada com êxito.
Códigos de resposta da série 400 (erros do cliente)	Não repetir	Há um dos seguintes problemas com a solicitação: <ul style="list-style-type: none"> Ela inclui um parâmetro ou uma combinação de parâmetros que não é válido. Ela usa uma ação ou um recurso para o qual você não tem permissões.

Resposta	Recomendação	Comentários
		<ul style="list-style-type: none"> Ela usa um recurso que está em processo de mudança de estado. <p>Se a solicitação envolver um recurso em processo de mudança de estado, a repetição da solicitação poderá ser bem-sucedida.</p>
Códigos de resposta da série 500 (erros do servidor)	Tentar novamente	O erro é causado por um problema no AWS servidor e geralmente é transitório. Repita a solicitação com uma estratégia de recuo apropriada.

Limitação de solicitações para a Amazon EC2 API

A EC2 Amazon limita as EC2 API solicitações para cada AWS conta por região. Fazemos isso para ajudar no desempenho do serviço e para garantir o uso justo para todos os EC2 clientes da Amazon. A limitação garante que as solicitações para a Amazon EC2 API não excedam os limites máximos de API solicitação permitidos. APIs solicitações estão sujeitas aos limites de solicitação, independentemente de serem originárias de:

- Aplicativos de terceiros
- Uma ferramentas da linha de comando
- O EC2 console da Amazon

Se você exceder um limite de API limitação, receberá o código de RequestLimitExceeded erro.

Conteúdo

- [Como o controle de utilização é aplicado](#)
- [Limites de controle de utilização](#)
- [Limitação do API monitor](#)
- [Tentativas repetidas e recuo exponencial](#)

- [Solicitar um aumento de limite de](#)

Como o controle de utilização é aplicado

A Amazon EC2 usa o [algoritmo de token bucket](#) para implementar a API limitação. Com esse algoritmo, sua conta tem um bucket que contém um número específico de tokens. O número de tokens no bucket representa seu limite de limitação a qualquer segundo.

A Amazon EC2 implementa dois tipos de API limitação:

API tipos de limitação

- [Limitação de intervalo de solicitações](#)
- [Limitação da taxa de recursos](#)

Limitação de intervalo de solicitações

Com a limitação da taxa de solicitação, cada uma API é avaliada individualmente e você é limitado pelo número de solicitações feitas por vez. Cada solicitação que você faz remove um token do bucket. Por exemplo, o tamanho do token bucket para `DescribeHosts`, uma ação sem API mutação, é de 100 tokens. Você pode fazer até 100 `DescribeHosts` solicitações em um segundo. Se você exceder 100 solicitações em um segundo, isso será limitado e as solicitações restantes nesse segundo falharão; no entanto, as solicitações de outras não API serão afetadas.

Os buckets são recarregados automaticamente a uma taxa definida. Se o bucket estiver abaixo de sua capacidade máxima, um determinado número de tokens será adicionado a ele a cada segundo até atingir sua capacidade máxima. Se o balde estiver cheio quando os tokens de recarga chegarem, eles serão descartados. O bucket não pode conter mais do que seu número máximo de tokens. Por exemplo, o tamanho do bucket para `DescribeHosts`, uma ação sem API mutação, é de 100 tokens e a taxa de recarga é de 20 tokens por segundo. Se você fizer 100 `DescribeHosts` solicitações em um segundo, o bucket será reduzido para zero (0) tokens. O balde é então reabastecido com 20 fichas a cada segundo, até atingir sua capacidade máxima de 100 fichas. Isso significa que um bucket vazio atinge sua capacidade máxima após 5 segundos se nenhuma solicitação for feita durante esse período.

Você não precisa esperar que o bucket esteja completamente cheio para poder fazer API solicitações. Você pode usar fichas de recarga à medida que elas são adicionadas ao balde. Se você usar imediatamente os tokens de recarga, o balde não atingirá sua capacidade máxima. Por exemplo, o tamanho do bucket para `DescribeHosts`, uma ação sem API mutação, é de 100

tokens e a taxa de recarga é de 20 tokens por segundo. Se você esgotar o bucket fazendo 100 API solicitações em um segundo, poderá continuar fazendo 20 API solicitações por segundo usando os tokens de recarga à medida que forem adicionados ao bucket. O bucket pode ser reabastecido até a capacidade máxima somente se você fizer menos de 20 API solicitações por segundo.

Limitação da taxa de recursos

Algumas API ações, como `RunInstances` e `TerminateInstances`, conforme descrito na tabela a seguir, usam a limitação da taxa de recursos além da limitação da taxa de solicitação. Essas API ações têm um repositório de tokens de recursos separado que se esgota com base no número de recursos afetados pela solicitação. Assim como os buckets de tokens de solicitação, os buckets de tokens de recursos têm um intervalo máximo que permite que você estoure e uma taxa de recarga que permite manter uma taxa constante de solicitações pelo tempo que for necessário. Se você exceder um limite de intervalo específico para um API, inclusive quando um compartimento ainda não tiver sido reabastecido para suportar a próxima API solicitação, a ação do API será limitada, mesmo que você não tenha atingido o limite total de API aceleração.

Por exemplo, o tamanho do bucket de tokens de recursos `RunInstances` é de 1000 tokens e a taxa de recarga é de dois tokens por segundo. Portanto, você pode iniciar imediatamente 1.000 instâncias usando qualquer número de API solicitações, como uma solicitação para 1.000 instâncias ou quatro solicitações para 250 instâncias. Depois que o bucket de tokens de recursos estiver vazio, você poderá iniciar até duas instâncias por segundo, usando uma solicitação para duas instâncias ou duas solicitações para uma instância.

Para obter mais informações, consulte [Tamanhos de compartimentos de tokens de recursos e taxas de recarga](#).

Limites de controle de utilização

As seções a seguir descrevem os tamanhos do bucket de tokens de solicitação e do bucket de tokens de recursos e as taxas de recarga.

Limites

- [Solicite tamanhos de baldes de tokens e taxas de recarga](#)
- [Tamanhos de compartimentos de tokens de recursos e taxas de recarga](#)

Solicite tamanhos de baldes de tokens e taxas de recarga

Para fins de limitação da taxa de solicitação, API as ações são agrupadas nas seguintes categorias:

- **Ações não mutantes** — API ações que recuperam dados sobre recursos. Essa categoria geralmente inclui todas as `Describe*`, `List*Search*`, e `Get*` API ações `DescribeRouteTables`, como `SearchTransitGatewayRoutes`, `GetIpamPoolCidrs` e. Essas API ações geralmente têm os maiores limites de API limitação.
- **[Ações não filtradas e não paginadas sem mutação](#)** — Um subconjunto específico de ações sem mutação que, quando API solicitadas sem especificar paginação ou filtro, usam tokens de um conjunto de tokens menor. É recomendável que você use paginação e filtragem para que os tokens sejam deduzidos do repositório de tokens padrão (maior).
- **Ações mutantes** — API ações que criam, modificam ou excluem recursos. Essa categoria geralmente inclui todas as API ações que não são categorizadas como ações não mutantes, como `AllocateHosts`, e `ModifyHosts` `CreateCapacityReservation` Essas ações têm um limite de limitação menor do que as ações sem API mutação.
- **Ações que consomem muitos recursos** — ações mutantes API que levam mais tempo e consomem mais recursos para serem concluídas. Essas ações têm um limite de limitação ainda menor do que as ações de mutação. Eles são estrangulados separadamente de outras ações mutantes.
- **Ações sem mutação do console** — ações sem mutação que são API solicitadas no console da Amazon. EC2 Essas API ações são limitadas separadamente de outras ações não API mutantes.
- **Ações não categorizadas** — São API ações que recebem seus próprios tamanhos de pacotes de tokens e taxas de recarga, embora, por definição, se encaixem em uma das outras categorias.

A tabela a seguir mostra os tamanhos dos repositórios de tokens de solicitação e as taxas de recarga para todas as AWS regiões.

API categoria de ação	Ações	Capacidade máxima do bucket	Taxa de recarga do balde
Ações não mutantes	Todas as ações <code>Describe*</code> , <code>List*Search*</code> , e que não sejam <code>Get*</code> API ações não filtradas e não paginadas, não	100	20

API categoria de ação	Ações	Capacidade máxima do bucket	Taxa de recarga do balde
	mutantes ou ações não categorizadas.		
Ações não mutantes não filtradas e não paginadas	<ul style="list-style-type: none"> • DescribeInstances • DescribeInstanceStatus • DescribeNetworkInterfaces • DescribeSecurityGroups • DescribeSnapshots • DescribeSpotInstanceRequests • DescribeVolumes 	50	10

API categoria de ação	Ações	Capacidade máxima do bucket	Taxa de recarga do balde
Ações mutantes	Todas as ações mutantes que não são API ações que consomem muitos recursos ou ações não categorizadas.	50	5

API categoria de ação	Ações	Capacidade máxima do bucket	Taxa de recarga do balde
Ações que consomem muitos recursos	<ul style="list-style-type: none"> • AcceptVpcPeeringConnection • AuthorizeSecurityGroupIngress • CancelSpotInstanceRequests • CreateKeyPair • CreateVpcPeeringConnection • DeleteVpcPeeringConnection • RejectVpcPeeringConnection • RevokeSecurityGroupIngress • RequestSpotInstances 	50	5

API categoria de ação	Ações	Capacidade máxima do bucket	Taxa de recarga do balde
Ações não mutantes do console	<ul style="list-style-type: none"> Describe* Get* 	100	10
Ações não categorizadas	AcceptVpcEndpointConnections	10	1
	AdvertiseByoipCidr	1	0.1
	AssignIpv6Addresses	100	5
	AssignPrivateIpAddresses	100	5
	AssignPrivateNatGatewayAddress	10	1
	AssociateEnclaveCertificateIamRole	10	1
	AssociateIamInstanceProfile	100	5
	AssociateNatGatewayAddress	10	1

API categoria de ação	Ações	Capacidade máxima do bucket	Taxa de recarga do balde
	AttachVerifiedAccessTrustProvider	10	2
	CreateDefaultSubnet	1	1
	CreateDefaultVpc	1	1
	CopyImage	100	1
	CreateLaunchTemplateVersion	100	5
	CreateNatGateway	10	1
	CreateNetworkInterface	100	5
	CreateRestoreImageTask	50	0.1
	CreateSnapshot	100	5
	CreateSnapshots	100	5
	CreateStoreImageTask	50	0.1
	CreateTags	100	10

API categoria de ação	Ações	Capacidade máxima do bucket	Taxa de recarga do balde
	CreateVerifiedAccessEndpoint	20	4
	CreateVerifiedAccessGroup	10	2
	CreateVerifiedAccessInstance	10	2
	CreateVerifiedAccessTrustProvider	10	2
	CreateVolume	100	5
	CreateVpcEndpoint	4	0.3
	CreateVpcEndpointServiceConfiguration	10	1
	DeleteNatGateway	10	1
	DeleteNetworkInterface	100	5
	DeleteSnapshot	100	5
	DeleteTags	100	10

API categoria de ação	Ações	Capacidade máxima do bucket	Taxa de recarga do balde
	DeleteQueuedReservedInstances	5	5
	DeleteVerifiedAccessEndpoint	20	4
	DeleteVerifiedAccessGroup	10	2
	DeleteVerifiedAccessInstance	10	2
	DeleteVerifiedAccessTrustProvider	10	2
	DeleteVolume	100	5
	DeleteVpcEndpoints	4	0.3
	DeleteVpcEndpointServiceConfigurations	10	1
	DeprovisionByoipCidr	1	0.1
	DeregisterImage	100	5

API categoria de ação	Ações	Capacidade máxima do bucket	Taxa de recarga do balde
	DetachVerifiedAccessTrustProvider	10	2
	DescribeByoipCidrs	1	0,5
	DescribeCapacityBlockOfferings	10	0,15
	DescribeInstanceTopology	1	1
	DescribeMovingAddresses	1	1
	DescribeReservedInstancesOfferings	10	10
	DescribeSpotFleetRequestHistory	100	5
	DescribeSpotFleetInstances	100	5
	DescribeSpotFleetRequests	50	3

API categoria de ação	Ações	Capacidade máxima do bucket	Taxa de recarga do balde
	DescribeStoreImageTasks	50	0,5
	DescribeVerifiedAccessInstanceLoggingConfigurations	10	2
	DisableFastLaunch	5	2
	DisableImageBlockPublicAccess	1	0,1
	DisableSnapshotBlockPublicAccess	1	0.1
	DisassociateEnclaveCertificateIamRole	10	1
	DisassociateIamInstanceProfile	100	5
	DisassociateNatGatewayAddress	10	1

API categoria de ação	Ações	Capacidade máxima do bucket	Taxa de recarga do balde
	EnableFastLaunch	5	2
	EnableImageBlockPublicAccess	1	0,1
	EnableSnapshotBlockPublicAccess	1	0.1
	GetAssociatedEnclaveCertificateIamRoles	10	1
	ModifyImageAttribute	100	5
	ModifyInstanceMetadataOptions	100	5
	ModifyLaunchTemplate	100	5
	ModifyNetworkInterfaceAttribute	100	5
	ModifySnapshotAttribute	100	5
	ModifyVerifiedAccessEndpoint	20	4

API categoria de ação	Ações	Capacidade máxima do bucket	Taxa de recarga do balde
	ModifyVerifiedAccessEndpointPolicy	20	4
	ModifyVerifiedAccessGroup	10	2
	ModifyVerifiedAccessGroupPolicy	20	4
	ModifyVerifiedAccessInstance	10	2
	ModifyVerifiedAccessInstanceLoggingConfiguration	10	2
	ModifyVerifiedAccessTrustProvider	10	2
	ModifyVpcEndpoint	4	0.3
	ModifyVpcEndpointServiceConfiguration	10	1

API categoria de ação	Ações	Capacidade máxima do bucket	Taxa de recarga do balde
	MoveAddressesToVpc	1	1
	ProvisionByoipCidr	1	0.1
	PurchaseCapacityBlock	10	0,15
	PurchaseReservedInstancesOffering	5	5
	RejectVpcEndpointConnections	10	1
	RestoreAddressToClassic	1	1
	RunInstances	5	2
	StartInstances	5	2
	TerminateInstances	100	5
	UnassignPrivateIpAddresses	100	5
	UnassignPrivateNatGatewayAddress	10	1

API categoria de ação	Ações	Capacidade máxima do bucket	Taxa de recarga do balde
	WithdrawB yoipCidr	1	0.1

Tamanhos de compartimentos de tokens de recursos e taxas de recarga

A tabela a seguir lista os tamanhos dos buckets de tokens de recursos e as taxas de recarga para API ações que usam limitação de taxa de recursos.

API ação	Capacidade máxima do bucket	Taxa de recarga do balde
RunInstances	1000	2
TerminateInstances	1000	20
StartInstances	1000	2
StopInstances	1000	20

Limitação do API monitor

Você pode usar CloudWatch a Amazon para monitorar suas EC2 API solicitações da Amazon e coletar e rastrear métricas sobre a API limitação. Você também pode criar um alarme para avisá-lo quando estiver perto de atingir os limites de API limitação. Para obter mais informações, consulte [Monitore EC2 API solicitações da Amazon usando a Amazon CloudWatch](#).

Tentativas repetidas e recuo exponencial

Talvez seu aplicativo precise repetir uma API solicitação. Por exemplo:

- Para verificar se há uma atualização no status de um recurso
- Para enumerar um grande número de recursos (por exemplo, todos os seus volumes)
- Para repetir uma solicitação depois que ela falhar com um erro do servidor (5xx) ou um erro de limitação

No entanto, para um erro do cliente (4xx), você deve revisar a solicitação para corrigir o problema antes de tentar a solicitação novamente.

Alterações no status do recurso

Antes de iniciar a pesquisa para verificar se há atualizações de status, dê tempo para que a solicitação seja potencialmente concluída. Por exemplo, aguarde alguns minutos antes de verificar se sua instância está ativa. Ao iniciar a pesquisa, use um intervalo de espera apropriado entre solicitações sucessivas para diminuir a taxa de API solicitações. Para obter os melhores resultados, use um intervalo de latência crescente ou variável.

Como alternativa, você pode usar EventBridge a Amazon para notificá-lo sobre o status de alguns recursos. Por exemplo, você pode usar o evento EC2Instance State-Change Notification para notificá-lo sobre uma mudança de estado em uma instância. Para obter mais informações, consulte [Automatizar a Amazon EC2 usando EventBridge](#).

Repetições

Quando você precisar pesquisar ou repetir uma API solicitação, recomendamos usar um algoritmo de recuo exponencial para calcular o intervalo de espera entre as solicitações. A ideia por trás do recuo exponencial é usar esperas progressivamente mais longas entre as novas tentativas para respostas de erro consecutivas. Você deve implementar um intervalo máximo de atraso, bem como um número máximo de novas tentativas. Você também pode usar o jitter (atraso aleatório) para evitar colisões sucessivas. Para obter mais informações, consulte [Tempos limite, novas tentativas e recuo com variação de sinal](#).

Cada um AWS SDK implementa a lógica de repetição automática. Para obter mais informações, consulte o [comportamento de repetição](#) no Guia de referência de ferramentas AWS SDKs e ferramentas.

Solicitar um aumento de limite de

Você pode solicitar um aumento nos limites de API limitação do seu. Conta da AWS

Para solicitar acesso a esse recurso

1. [AWS Support Centro](#) aberto.
2. Escolha Criar caso.
3. Escolha Conta e faturamento.

4. Em Serviço, escolha Informações gerais e Introdução.
5. Em Categoria, escolha Uso AWS e serviços.
6. Selecione Próxima etapa: informações adicionais.
7. Em Subject (Assunto), insira **Request an increase in my Amazon EC2 API throttling limits**.
8. Em Descrição, insira **Please increase the API throttling limits for my account. Related page: <https://docs.aws.amazon.com/AWSEC2/latest/APIReference/throttling.html>**. Inclua também as seguintes informações:
 - Uma descrição do tipo de caso de uso.
 - As regiões em que você precisa de um aumento.
 - A janela de uma hora, emUTC, quando ocorreu o pico de limitação ou uso (para calcular o novo limite de limitação).
9. Escolha Próxima etapa: solucione ou entre em contato conosco.
10. Na guia Fale conosco, escolha seu idioma de contato preferido e método de contato.
11. Selecione Enviar.

Crie recursos do Amazon EC2 usando o AWS CLI

Você pode criar e gerenciar seus recursos do Amazon EC2 usando o AWS Command Line Interface (AWS CLI) em um shell de linha de comando. AWS CLI Ele fornece acesso direto às APIs do Serviços da AWS, como o Amazon EC2.

Para obter a sintaxe e exemplos dos comandos para o Amazon EC2, [consulte](#) ec2 na AWS CLI Referência de comandos. Você também pode encontrar esses exemplos em [aws-cli/awscli/examples/ec2](#) no github.

Saiba mais sobre o AWS CLI

Para saber mais sobre o AWS CLI, consulte os seguintes recursos:

- [AWS Command Line Interface](#)
- [AWS Command Line Interface Guia do usuário para a versão 2](#)
- [AWS Command Line Interface Guia do usuário para a versão 1](#)

Crie recursos do Amazon EC2 usando AWS CloudFormation

O Amazon EC2 é integrado com AWS CloudFormation, um serviço que ajuda você a modelar e configurar seus AWS recursos para que você possa gastar menos tempo criando e gerenciando seus recursos e infraestrutura. Você cria um modelo que descreve os AWS recursos necessários (como instâncias e sub-redes) e AWS CloudFormation provisiona e configura esses recursos para você.

Ao usar AWS CloudFormation, você pode reutilizar seu modelo para configurar seus recursos do Amazon EC2 de forma consistente e repetida. Descreva seus recursos uma vez e, em seguida, provisione os mesmos recursos repetidamente em várias Contas da AWS regiões.

Amazon EC2 e modelos AWS CloudFormation

[Para provisionar e configurar recursos para o Amazon EC2 e serviços relacionados, você deve entender AWS CloudFormation os modelos.](#) Os modelos são arquivos de texto formatados em JSON ou YAML. Esses modelos descrevem os recursos que você provisionará em suas AWS CloudFormation pilhas. Se você não estiver familiarizado com JSON ou YAML, você pode usar o AWS CloudFormation Designer para ajudá-lo a começar a usar modelos. AWS CloudFormation Para obter mais informações, consulte [O que é AWS CloudFormation Designer?](#) no Guia do AWS CloudFormation usuário.

Recursos para o Amazon EC2

Recursos de computação

- [AWS::EC2::CapacityReservation](#)
- [AWS::EC2::CapacityReservationFrota](#)
- [AWS::EC2::EC2Fleet](#)
- [AWS::EC2::EC2Fleet](#)
- [AWS::EC2::Host](#)
- [AWS::EC2::Instance](#)
- [AWS::EC2::InstanceConnectPonto final](#)
- [AWS::EC2::LaunchTemplate](#)
- [AWS::EC2::PlacementGroup](#)

- [AWS::EC2::SpotFleet](#)

Recursos de redes

- [AWS::EC2::CarrierGateway](#)
- [AWS::EC2::ClientVpnAuthorizationRule](#)
- [AWS::EC2::ClientVpnPonto final](#)
- [AWS::EC2::ClientVpnRota](#)
- [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::LocalGatewayRota](#)
- [AWS::EC2::LocalGatewayRouteTable](#)
- [AWS::EC2::LocalGatewayRouteTableVirtualInterfaceGroupAssociation](#)
- [AWS::EC2::LocalGatewayRouteTableAssociação VPC](#)
- [AWS::EC2::NatGateway](#)
- [AWS::EC2::NetworkInterface](#)
- [AWS::EC2::NetworkInsightsAccessScope](#)

- [AWS::EC2::NetworkInsightsAccessScopeAnalysis](#)
- [AWS::EC2::NetworkInsightsAnálise](#)
- [AWS::EC2::NetworkInsightsCaminho](#)
- [AWS::EC2::NetworkInterfaceAnexo](#)
- [AWS::EC2::NetworkInterfacePermissão](#)
- [AWS::EC2::NetworkPerformanceMetricSubscription](#)
- [AWS::EC2::PrefixList](#)
- [AWS::EC2::Route](#)
- [AWS::EC2::RouteTable](#)
- [AWS::EC2::Subnet](#)
- [AWS::EC2::SubnetCidrBloquear](#)
- [AWS::EC2::SubnetNetworkAclAssociation](#)
- [AWS::EC2::SubnetRouteTableAssociation](#)
- [AWS::EC2::TrafficMirrorFiltrar](#)
- [AWS::EC2::TrafficMirrorFilterRule](#)
- [AWS::EC2::TrafficMirrorSessão](#)
- [AWS::EC2::TrafficMirrorAlvo](#)
- [AWS::EC2::TransitGateway](#)
- [AWS::EC2::TransitGatewayAnexo](#)
- [AWS::EC2::TransitGatewayConnect](#)
- [AWS::EC2::TransitGatewayMulticastDomain](#)
- [AWS::EC2::TransitGatewayMulticastDomainAssociation](#)
- [AWS::EC2::TransitGatewayMulticastGroupMember](#)
- [AWS::EC2::TransitGatewayMulticastGroupSource](#)
- [AWS::EC2::TransitGatewayPeeringAttachment](#)
- [AWS::EC2::TransitGatewayRota](#)
- [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](#)

Recursos de segurança

- [AWS::EC2::KeyPair](#)
- [AWS::EC2::NetworkAcl](#)
- [AWS::EC2::NetworkAclEntrada](#)
- [AWS::EC2::SecurityGroup](#)
- [AWS::EC2::SecurityGroupSaída](#)
- [AWS::EC2::SecurityGroupEntrada](#)
- [AWS::EC2::VerifiedAccessPonto final](#)
- [AWS::EC2::VerifiedAccessGrupo](#)
- [AWS::EC2::VerifiedAccessInstância](#)
- [AWS::EC2::VerifiedAccessTrustProvider](#)

Recursos de armazenamento

- [AWS::EC2::SnapshotBlockPublicAccess](#)
- [AWS::EC2::Volume](#)
- [AWS::EC2::VolumeAttachment](#)

Saiba mais sobre AWS CloudFormation

Para saber mais sobre isso AWS CloudFormation, consulte os seguintes recursos:

- [AWS CloudFormation](#)
- [AWS CloudFormation Guia do usuário](#)

Crie EC2 recursos da Amazon usando um AWS SDK

AWS fornece kits de desenvolvimento de software (SDK) para muitas linguagens de programação populares. E SDK torna o desenvolvimento mais eficiente ao fornecer o seguinte:

- Componentes e bibliotecas pré-criados que você pode incorporar aos seus aplicativos
- Ferramentas específicas de linguagem, como compiladores e depuradores
- Assinatura criptográfica de solicitações de serviço
- Solicitar novas tentativas
- Tratamento da resposta de erro

Exemplos de código para a Amazon EC2 API

Os exemplos de código fornecidos por AWS mostram como usar um API e realizar tarefas específicas. Para obter exemplos para a Amazon EC2API, consulte [Exemplos de códigos para a Amazon EC2](#). Para obter exemplos adicionais, consulte [Encontre exemplos de código para o AWS SDKs](#) ou [aws-doc-sdk-examples](#) no github.

Saiba mais sobre o AWS SDKs

Para saber mais sobre o AWS SDKs, consulte os seguintes recursos:

- [AWS SDKs Guia de referência de ferramentas e ferramentas](#)
- [Ferramentas para construir AWS](#)
- [O que é um SDK?](#)

API de baixo nível para Amazon EC2

A API de baixo nível para o Amazon EC2 é a interface em nível de protocolo para o Amazon EC2. Ao usar a API de baixo nível, você deve formatar cada solicitação HTTPS corretamente e adicionar uma assinatura digital válida a cada solicitação. Para obter mais informações, consulte [Fazer solicitações para a API do Amazon EC2 na Referência da API](#) do Amazon EC2. Como alternativa, você pode usar um AWS SDK, que constrói e assina as solicitações em seu nome. Para ter mais informações, consulte [Usando um AWS SDK](#).

A API do Amazon EC2 consiste em ações e tipos de dados para vários serviços. Para visualizar as ações de cada serviço, consulte as seguintes páginas na Referência de API do Amazon EC2.

- [AWS Client VPN actions](#)
- [Ações do Amazon EBS](#)
- [Ações do Amazon EC2](#)
- [AWS Network Manager actions](#)
- [AWS Ações do Nitro Enclaves](#)
- [AWS Outposts actions](#)
- [AWS PrivateLink actions](#)
- [Ações da lixeira](#)
- [AWS Site-to-Site VPN actions](#)
- [AWS Transit Gateway actions](#)
- [Acesso Verificado pela AWS actions](#)
- [Ações de importação/exportação da VM](#)
- [Ações da Amazon VPC](#)
- [Ações IPAM da Amazon VPC](#)
- [AWS Wavelength actions](#)

Use Console-to-Code para gerar código para as ações EC2 do console

O console oferece um caminho guiado para criar recursos e testar protótipos. Se você quiser criar os mesmos recursos em grande escala, precisará de um código de automação. Console-to-Code é um recurso do Amazon Q Developer que pode ajudar você a começar a usar seu código de automação. Console-to-Code registra as ações do console, incluindo configurações padrão e parâmetros compatíveis. Em seguida, ele usa IA generativa para sugerir código no formato de sua preferência *infrastructure-as-code* (IaC) para as ações que você deseja. Como o fluxo de trabalho do console garante que os valores dos parâmetros que você especifica sejam válidos juntos, o código que você gera usando Console-to-Code tem valores de parâmetros compatíveis. Você pode usar o código como ponto de partida e, depois, personalizá-lo para deixá-lo pronto para produção no seu caso de uso específico.

Por exemplo, com Console-to-Code, você pode gravar o lançamento de uma EC2 instância da Amazon e optar por gerar código no AWS CloudFormation JSON formato. Em seguida, você pode copiar esse código e personalizá-lo para uso em seu modelo do AWS CloudFormation .

Console-to-Code atualmente pode gerar *infrastructure-as-code* (IaC) nos seguintes idiomas e formatos:

- CDKJava
- CDKPython
- CDK TypeScript
- CloudFormation JSON
- CloudFormation YAML

Para obter mais informações e instruções sobre como usar Console-to-Code, consulte [Automação de AWS serviços com o Amazon Q Developer Console-to-Code no Amazon Q Developer User Guide](#).

Exemplos de código para a Amazon EC2 usando AWS SDKs

Os exemplos de código a seguir mostram como usar a Amazon EC2 com um kit AWS de desenvolvimento de software (SDK).

As noções básicas são exemplos de código que mostram como realizar as operações essenciais em um serviço.

Ações são trechos de código de programas maiores e devem ser executadas em contexto. Embora as ações mostrem como chamar funções de serviço individuais, você pode ver as ações no contexto em seus cenários relacionados.

Os cenários são exemplos de código que mostram como realizar tarefas específicas chamando várias funções dentro de um serviço ou combinadas com outros Serviços da AWS.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Conceitos básicos

Olá Amazon EC2

Os exemplos de código a seguir mostram como começar a usar a AmazonEC2.

.NET

AWS SDK for .NET

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
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>Async task.</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>();

        try
        {
            // Retrieve information for up to 10 Amazon EC2 security groups.
            var request = new DescribeSecurityGroupsRequest { MaxResults = 10, };
            var securityGroups = new List<SecurityGroup>();

            var paginatorForSecurityGroups =
                ec2Client.Paginators.DescribeSecurityGroups(request);

            await foreach (var securityGroup in
                paginatorForSecurityGroups.SecurityGroups)
            {
                securityGroups.Add(securityGroup);
            }

            // Now print the security groups returned by the call to
            // DescribeSecurityGroupsAsync.
            Console.WriteLine("Welcome to the EC2 Hello Service example. " +
                "\nLet's list your Security Groups:");
            securityGroups.ForEach(group =>
            {
                Console.WriteLine(
                    $"Security group: {group.GroupName} ID: {group.GroupId}");
            });
        }
    }
}
```

```
    });  
  }  
  catch (AmazonEC2Exception ex)  
  {  
    Console.WriteLine($"An Amazon EC2 service error occurred while  
listing security groups. {ex.Message}");  
  }  
  catch (Exception ex)  
  {  
    Console.WriteLine($"An error occurred while listing security groups.  
{ex.Message}");  
  }  
  }  
}
```

- Para API obter detalhes, consulte [DescribeSecurityGroup](#) sem AWS SDK for .NET API Referência.

C++

SDK para C++

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

Código para o CMakeLists arquivo.txtCMake.

```
# 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})
```

Código para o arquivo fonte hello_ec2.cpp.

```
#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) {
    (void)argc;
    (void)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) {
            Aws::EC2::Model::DescribeInstancesOutcome 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;
}

```

- Para API obter detalhes, consulte [DescribeSecurityGroup](#) em AWS SDK for C++ API Referência.

Java

SDK para Java 2.x

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```

/**
 * Asynchronously describes the security groups for the specified group ID.
 *
 * @param groupName the name of the security group to describe
 * @return a {@link CompletableFuture} that represents the asynchronous
operation
 *         of describing the security groups. The future will complete with a

```

```
*      {@link DescribeSecurityGroupsResponse} object that contains the
*      security group information.
*/
public CompletableFuture<String> describeSecurityGroupArnByNameAsync(String
groupName) {
    DescribeSecurityGroupsRequest request =
DescribeSecurityGroupsRequest.builder()
        .groupNames(groupName)
        .build();

    DescribeSecurityGroupsPublisher paginator =
getAsyncClient().describeSecurityGroupsPaginator(request);
    AtomicReference<String> groupIdRef = new AtomicReference<>();
    return paginator.subscribe(response -> {
        response.securityGroups().stream()
            .filter(securityGroup ->
securityGroup.groupName().equals(groupName))
            .findFirst()
            .ifPresent(securityGroup ->
groupIdRef.set(securityGroup.groupId()));
    }).thenApply(v -> {
        String groupId = groupIdRef.get();
        if (groupId == null) {
            throw new RuntimeException("No security group found with the
name: " + groupName);
        }
        return groupId;
    }).exceptionally(ex -> {
        logger.info("Failed to describe security group: " + ex.getMessage());
        throw new RuntimeException("Failed to describe security group", ex);
    });
}
```

- Para API obter detalhes, consulte [DescribeSecurityGroup](#) sem AWS SDK for Java 2.x API Referência.

JavaScript

SDK para JavaScript (v3)

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
import { DescribeSecurityGroupsCommand, EC2Client } from "@aws-sdk/client-ec2";

// Call DescribeSecurityGroups and display the result.
export const main = async () => {
  const client = new EC2Client();
  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);
  }
};

// Call function if run directly.
import { fileURLToPath } from "node:url";
if (process.argv[1] === fileURLToPath(import.meta.url)) {
  main();
}
```

- Para API obter detalhes, consulte [DescribeSecurityGroups](#) em AWS SDK for JavaScript API Referência.

Kotlin

SDK para Kotlin

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
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}")
        }
    }
}
```

- Para API obter detalhes, consulte a [DescribeSecurityGroups](#) referência AWS SDK do Kotlin API.

Python

SDK para Python (Boto3)

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
def hello_ec2(ec2_client):
    """
    Use the AWS SDK for Python (Boto3) to list the security groups in your
    account.
    This example uses the default settings specified in your shared credentials
    and config files.

    :param ec2_client: A Boto3 EC2 client. This client provides low-level
        access to AWS EC2 services.
    """
    print("Hello, Amazon EC2! Let's list up to 10 of your security groups:")
    try:
        paginator = ec2_client.get_paginator("describe_security_groups")
        response_iterator = paginator.paginate(MaxResults=10)
        for page in response_iterator:
            for sg in page["SecurityGroups"]:
                logger.info(f"\t{sg['GroupId']}: {sg['GroupName']}")
    except ClientError as err:
        logger.error("Failed to list security groups.")
        if err.response["Error"]["Code"] == "AccessDeniedException":
            logger.error("You do not have permission to list security groups.")
        raise

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

- Para API obter detalhes, consulte a [DescribeSecurityGroups](#) Referência AWS SDK do Python (Boto3). API

Ruby

SDK para Ruby

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
require 'aws-sdk-ec2'
require 'logger'

# EC2Manager is a class responsible for managing EC2 operations
# such as listing all EC2 instances in the current AWS account.
class EC2Manager
  def initialize(client)
    @client = client
    @logger = Logger.new($stdout)
  end

  # Lists and prints all EC2 instances in the current AWS account.
  def list_instances
    @logger.info('Listing instances')

    instances = fetch_instances

    if instances.empty?
      @logger.info('You have no instances')
    else
      print_instances(instances)
    end
  end

  private

  # Fetches all EC2 instances using pagination.
  #
  # @return [Array<Aws::EC2::Types::Instance>] List of EC2 instances.
  def fetch_instances
    paginator = @client.describe_instances
```

```
instances = []

paginator.each_page do |page|
  page.reservations.each do |reservation|
    reservation.instances.each do |instance|
      instances << instance
    end
  end
end

instances

end

# Prints details of the given EC2 instances.
#
# @param instances [Array<Aws::EC2::Types::Instance>] List of EC2 instances to
print.
def print_instances(instances)
  instances.each do |instance|
    @logger.info("Instance ID: #{instance.instance_id}")
    @logger.info("Instance Type: #{instance.instance_type}")
    @logger.info("Public IP: #{instance.public_ip_address}")
    @logger.info("Public DNS Name: #{instance.public_dns_name}")
    @logger.info("\n")
  end
end
end

if $PROGRAM_NAME == __FILE__
  ec2_client = Aws::EC2::Client.new(region: 'us-west-2')
  manager = EC2Manager.new(ec2_client)
  manager.list_instances
end
```

- Para API obter detalhes, consulte [DescribeSecurityGroups](#) em AWS SDK for Ruby API Referência.

Rust

SDK para Rust

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
async fn show_security_groups(client: &aws_sdk_ec2::Client, group_ids:
Vec<String>) {
    let response = client
        .describe_security_groups()
        .set_group_ids(Some(group_ids))
        .send()
        .await;

    match response {
        Ok(output) => {
            for group in output.security_groups() {
                println!(
                    "Found Security Group {} ({}), vpc id {} and description {}",
                    group.group_name().unwrap_or("unknown"),
                    group.group_id().unwrap_or("id-unknown"),
                    group.vpc_id().unwrap_or("vpcid-unknown"),
                    group.description().unwrap_or("(none)")
                );
            }
        }
        Err(err) => {
            let err = err.into_service_error();
            let meta = err.meta();
            let message = meta.message().unwrap_or("unknown");
            let code = meta.code().unwrap_or("unknown");
            eprintln!("Error listing EC2 Security Groups: ({code}) {message}");
        }
    }
}
```


- Para API obter detalhes, consulte [DescribeSecurityGroups](#) a AWS SDK API referência do Rust.

Exemplos de código

- [Exemplos básicos para o EC2 uso da Amazon AWS SDKs](#)
 - [Olá Amazon EC2](#)
 - [Aprenda o básico da Amazon EC2 com um AWS SDK](#)
 - [Ações para a Amazon EC2 usando AWS SDKs](#)
 - [Use AcceptVpcPeeringConnection com um CLI](#)
 - [Use AllocateAddress com um AWS SDK ou CLI](#)
 - [Use AllocateHosts com um CLI](#)
 - [Use AssignPrivateIpAddresses com um CLI](#)
 - [Use AssociateAddress com um AWS SDK ou CLI](#)
 - [Use AssociateDhcpOptions com um CLI](#)
 - [Use AssociateRouteTable com um CLI](#)
 - [Use AttachInternetGateway com um CLI](#)
 - [Use AttachNetworkInterface com um CLI](#)
 - [Use AttachVolume com um CLI](#)
 - [Use AttachVpnGateway com um CLI](#)
 - [Use AuthorizeSecurityGroupEgress com um CLI](#)
 - [Use AuthorizeSecurityGroupIngress com um AWS SDK ou CLI](#)
 - [Use CancelCapacityReservation com um CLI](#)
 - [Use CancelImportTask com um CLI](#)
 - [Use CancelSpotFleetRequests com um CLI](#)
 - [Use CancelSpotInstanceRequests com um CLI](#)
 - [Use ConfirmProductInstance com um CLI](#)
 - [Use CopyImage com um CLI](#)
 - [Use CopySnapshot com um CLI](#)
 - [Use CreateCapacityReservation com um CLI](#)
 - [Use CreateCustomerGateway com um CLI](#)
 - [Use CreateDhcpOptions com um CLI](#)

- [Use CreateFlowLogs com um CLI](#)
- [Use CreateImage com um CLI](#)
- [Use CreateInstanceExportTask com um CLI](#)
- [Use CreateInternetGateway com um CLI](#)
- [Use CreateKeyPair com um AWS SDK ou CLI](#)
- [Use CreateLaunchTemplate com um AWS SDK ou CLI](#)
- [Use CreateNetworkAcl com um CLI](#)
- [Use CreateNetworkAclEntry com um CLI](#)
- [Use CreateNetworkInterface com um CLI](#)
- [Use CreatePlacementGroup com um CLI](#)
- [Use CreateRoute com um CLI](#)
- [Use CreateRouteTable com um AWS SDK ou CLI](#)
- [Use CreateSecurityGroup com um AWS SDK ou CLI](#)
- [Use CreateSnapshot com um CLI](#)
- [Use CreateSpotDatafeedSubscription com um CLI](#)
- [Use CreateSubnet com um AWS SDK ou CLI](#)
- [Use CreateTags com um AWS SDK ou CLI](#)
- [Use CreateVolume com um CLI](#)
- [Use CreateVpc com um AWS SDK ou CLI](#)
- [Use CreateVpcEndpoint com um AWS SDK ou CLI](#)
- [Use CreateVpnConnection com um CLI](#)
- [Use CreateVpnConnectionRoute com um CLI](#)
- [Use CreateVpnGateway com um CLI](#)
- [Use DeleteCustomerGateway com um CLI](#)
- [Use DeleteDhcpOptions com um CLI](#)
- [Use DeleteFlowLogs com um CLI](#)
- [Use DeleteInternetGateway com um CLI](#)
- [Use DeleteKeyPair com um AWS SDK ou CLI](#)
- [Use DeleteLaunchTemplate com um AWS SDK ou CLI](#)
- [Use DeleteNetworkAcl com um CLI](#)

- [Use DeleteNetworkAclEntry com um CLI](#)
- [Use DeleteNetworkInterface com um CLI](#)
- [Use DeletePlacementGroup com um CLI](#)
- [Use DeleteRoute com um CLI](#)
- [Use DeleteRouteTable com um CLI](#)
- [Use DeleteSecurityGroup com um AWS SDK ou CLI](#)
- [Use DeleteSnapshot com um AWS SDK ou CLI](#)
- [Use DeleteSpotDatafeedSubscription com um CLI](#)
- [Use DeleteSubnet com um CLI](#)
- [Use DeleteTags com um CLI](#)
- [Use DeleteVolume com um CLI](#)
- [Use DeleteVpc com um AWS SDK ou CLI](#)
- [Use DeleteVpcEndpoint com um AWS SDK](#)
- [Use DeleteVpnConnection com um CLI](#)
- [Use DeleteVpnConnectionRoute com um CLI](#)
- [Use DeleteVpnGateway com um CLI](#)
- [Use DeregisterImage com um CLI](#)
- [Use DescribeAccountAttributes com um CLI](#)
- [Use DescribeAddresses com um AWS SDK ou CLI](#)
- [Use DescribeAvailabilityZones com um AWS SDK ou CLI](#)
- [Use DescribeBundleTasks com um CLI](#)
- [Use DescribeCapacityReservations com um CLI](#)
- [Use DescribeCustomerGateways com um CLI](#)
- [Use DescribeDhcpOptions com um CLI](#)
- [Use DescribeFlowLogs com um CLI](#)
- [Use DescribeHostReservationOfferings com um CLI](#)
- [Use DescribeHosts com um CLI](#)
- [Use DescribelamInstanceProfileAssociations com um AWS SDK ou CLI](#)
- [Use DescribeIdFormat com um CLI](#)
- [Use DescribeIdentityIdFormat com um CLI](#)

- [Use DescribeImageAttribute com um CLI](#)
- [Use DescribeImages com um AWS SDK ou CLI](#)
- [Use DescribeImportImageTasks com um CLI](#)
- [Use DescribeImportSnapshotTasks com um CLI](#)
- [Use DescribeInstanceAttribute com um CLI](#)
- [Use DescribeInstanceState com um AWS SDK ou CLI](#)
- [Use DescribeInstanceTypes com um AWS SDK ou CLI](#)
- [Use DescribeInstances com um AWS SDK ou CLI](#)
- [Use DescribeInternetGateways com um CLI](#)
- [Use DescribeKeyPairs com um AWS SDK ou CLI](#)
- [Use DescribeNetworkAcls com um CLI](#)
- [Use DescribeNetworkInterfaceAttribute com um CLI](#)
- [Use DescribeNetworkInterfaces com um CLI](#)
- [Use DescribePlacementGroups com um CLI](#)
- [Use DescribePrefixLists com um CLI](#)
- [Use DescribeRegions com um AWS SDK ou CLI](#)
- [Use DescribeRouteTables com um AWS SDK ou CLI](#)
- [Use DescribeScheduledInstanceAvailability com um CLI](#)
- [Use DescribeScheduledInstances com um CLI](#)
- [Use DescribeSecurityGroups com um AWS SDK ou CLI](#)
- [Use DescribeSnapshotAttribute com um CLI](#)
- [Use DescribeSnapshots com um AWS SDK ou CLI](#)
- [Use DescribeSpotDatafeedSubscription com um CLI](#)
- [Use DescribeSpotFleetInstances com um CLI](#)
- [Use DescribeSpotFleetRequestHistory com um CLI](#)
- [Use DescribeSpotFleetRequests com um CLI](#)
- [Use DescribeSpotInstanceRequests com um CLI](#)
- [Use DescribeSpotPriceHistory com um CLI](#)
- [Use DescribeSubnets com um AWS SDK ou CLI](#)
- [Use DescribeTags com um CLI](#)

- [Use DescribeVolumeAttribute com um CLI](#)
- [Use DescribeVolumeStatus com um CLI](#)
- [Use DescribeVolumes com um CLI](#)
- [Use DescribeVpcAttribute com um CLI](#)
- [Use DescribeVpcClassicLink com um CLI](#)
- [Use DescribeVpcClassicLinkDnsSupport com um CLI](#)
- [Use DescribeVpcEndpointServices com um CLI](#)
- [Use DescribeVpcEndpoints com um CLI](#)
- [Use DescribeVpcs com um AWS SDK ou CLI](#)
- [Use DescribeVpnConnections com um CLI](#)
- [Use DescribeVpnGateways com um CLI](#)
- [Use DetachInternetGateway com um CLI](#)
- [Use DetachNetworkInterface com um CLI](#)
- [Use DetachVolume com um CLI](#)
- [Use DetachVpnGateway com um CLI](#)
- [Use DisableVgwRoutePropagation com um CLI](#)
- [Use DisableVpcClassicLink com um CLI](#)
- [Use DisableVpcClassicLinkDnsSupport com um CLI](#)
- [Use DisassociateAddress com um AWS SDK ou CLI](#)
- [Use DisassociateRouteTable com um CLI](#)
- [Use EnableVgwRoutePropagation com um CLI](#)
- [Use EnableVolumelo com um CLI](#)
- [Use EnableVpcClassicLink com um CLI](#)
- [Use EnableVpcClassicLinkDnsSupport com um CLI](#)
- [Use GetConsoleOutput com um CLI](#)
- [Use GetHostReservationPurchasePreview com um CLI](#)
- [Use GetPasswordData com um AWS SDK ou CLI](#)
- [Use ImportImage com um CLI](#)
- [Use ImportKeyPair com um CLI](#)
- [Use ImportSnapshot com um CLI](#)

- [Use ModifyCapacityReservation com um CLI](#)
- [Use ModifyHosts com um CLI](#)
- [Use ModifyIdFormat com um CLI](#)
- [Use ModifyImageAttribute com um CLI](#)
- [Use ModifyInstanceAttribute com um CLI](#)
- [Use ModifyInstanceCreditSpecification com um CLI](#)
- [Use ModifyNetworkInterfaceAttribute com um CLI](#)
- [Use ModifyReservedInstances com um CLI](#)
- [Use ModifySnapshotAttribute com um CLI](#)
- [Use ModifySpotFleetRequest com um CLI](#)
- [Use ModifySubnetAttribute com um CLI](#)
- [Use ModifyVolumeAttribute com um CLI](#)
- [Use ModifyVpcAttribute com um CLI](#)
- [Use MonitorInstances com um AWS SDK ou CLI](#)
- [Use MoveAddressToVpc com um CLI](#)
- [Use PurchaseHostReservation com um CLI](#)
- [Use PurchaseScheduledInstances com um CLI](#)
- [Use RebootInstances com um AWS SDK ou CLI](#)
- [Use RegisterImage com um CLI](#)
- [Use RejectVpcPeeringConnection com um CLI](#)
- [Use ReleaseAddress com um AWS SDK ou CLI](#)
- [Use ReleaseHosts com um CLI](#)
- [Use ReplacelamInstanceProfileAssociation com um AWS SDK ou CLI](#)
- [Use ReplaceNetworkAclAssociation com um CLI](#)
- [Use ReplaceNetworkAclEntry com um CLI](#)
- [Use ReplaceRoute com um CLI](#)
- [Use ReplaceRouteTableAssociation com um CLI](#)
- [Use ReportInstanceStatus com um CLI](#)
- [Use RequestSpotFleet com um CLI](#)
- [Use RequestSpotInstances com um CLI](#)

- [Use ResetImageAttribute com um CLI](#)
 - [Use ResetInstanceAttribute com um CLI](#)
 - [Use ResetNetworkInterfaceAttribute com um CLI](#)
 - [Use ResetSnapshotAttribute com um CLI](#)
 - [Use RevokeSecurityGroupEgress com um CLI](#)
 - [Use RevokeSecurityGroupIngress com um CLI](#)
 - [Use RunInstances com um AWS SDK ou CLI](#)
 - [Use RunScheduledInstances com um CLI](#)
 - [Use StartInstances com um AWS SDK ou CLI](#)
 - [Use StopInstances com um AWS SDK ou CLI](#)
 - [Use TerminateInstances com um AWS SDK ou CLI](#)
 - [Use UnassignPrivateAddresses com um CLI](#)
 - [Use UnmonitorInstances com um AWS SDK ou CLI](#)
 - [Use UpdateSecurityGroupRuleDescriptionsIngress com um CLI](#)
- [Cenários para o EC2 uso da Amazon AWS SDKs](#)
 - [Crie e gerencie um serviço resiliente usando um AWS SDK](#)

Exemplos básicos para o EC2 uso da Amazon AWS SDKs

Os exemplos de código a seguir mostram como usar os conceitos básicos do Amazon Elastic Compute Cloud com AWS SDKs

Exemplos

- [Olá Amazon EC2](#)
- [Aprenda o básico da Amazon EC2 com um AWS SDK](#)
- [Ações para a Amazon EC2 usando AWS SDKs](#)
 - [Use AcceptVpcPeeringConnection com um CLI](#)
 - [Use AllocateAddress com um AWS SDK ou CLI](#)
 - [Use AllocateHosts com um CLI](#)
 - [Use AssignPrivateAddresses com um CLI](#)
 - [Use AssociateAddress com um AWS SDK ou CLI](#)

- [Use AssociateDhcpOptions com um CLI](#)
- [Use AssociateRouteTable com um CLI](#)
- [Use AttachInternetGateway com um CLI](#)
- [Use AttachNetworkInterface com um CLI](#)
- [Use AttachVolume com um CLI](#)
- [Use AttachVpnGateway com um CLI](#)
- [Use AuthorizeSecurityGroupEgress com um CLI](#)
- [Use AuthorizeSecurityGroupIngress com um AWS SDK ou CLI](#)
- [Use CancelCapacityReservation com um CLI](#)
- [Use CancellImportTask com um CLI](#)
- [Use CancelSpotFleetRequests com um CLI](#)
- [Use CancelSpotInstanceRequests com um CLI](#)
- [Use ConfirmProductInstance com um CLI](#)
- [Use CopyImage com um CLI](#)
- [Use CopySnapshot com um CLI](#)
- [Use CreateCapacityReservation com um CLI](#)
- [Use CreateCustomerGateway com um CLI](#)
- [Use CreateDhcpOptions com um CLI](#)
- [Use CreateFlowLogs com um CLI](#)
- [Use CreateImage com um CLI](#)
- [Use CreateInstanceExportTask com um CLI](#)
- [Use CreateInternetGateway com um CLI](#)
- [Use CreateKeyPair com um AWS SDK ou CLI](#)
- [Use CreateLaunchTemplate com um AWS SDK ou CLI](#)
- [Use CreateNetworkAcl com um CLI](#)
- [Use CreateNetworkAclEntry com um CLI](#)
- [Use CreateNetworkInterface com um CLI](#)
- [Use CreatePlacementGroup com um CLI](#)
- [Use CreateRoute com um CLI](#)
- [Use CreateRouteTable com um AWS SDK ou CLI](#)

- [Use CreateSecurityGroup com um AWS SDK ou CLI](#)
- [Use CreateSnapshot com um CLI](#)
- [Use CreateSpotDatafeedSubscription com um CLI](#)
- [Use CreateSubnet com um AWS SDK ou CLI](#)
- [Use CreateTags com um AWS SDK ou CLI](#)
- [Use CreateVolume com um CLI](#)
- [Use CreateVpc com um AWS SDK ou CLI](#)
- [Use CreateVpcEndpoint com um AWS SDK ou CLI](#)
- [Use CreateVpnConnection com um CLI](#)
- [Use CreateVpnConnectionRoute com um CLI](#)
- [Use CreateVpnGateway com um CLI](#)
- [Use DeleteCustomerGateway com um CLI](#)
- [Use DeleteDhcpOptions com um CLI](#)
- [Use DeleteFlowLogs com um CLI](#)
- [Use DeleteInternetGateway com um CLI](#)
- [Use DeleteKeyPair com um AWS SDK ou CLI](#)
- [Use DeleteLaunchTemplate com um AWS SDK ou CLI](#)
- [Use DeleteNetworkAcl com um CLI](#)
- [Use DeleteNetworkAclEntry com um CLI](#)
- [Use DeleteNetworkInterface com um CLI](#)
- [Use DeletePlacementGroup com um CLI](#)
- [Use DeleteRoute com um CLI](#)
- [Use DeleteRouteTable com um CLI](#)
- [Use DeleteSecurityGroup com um AWS SDK ou CLI](#)
- [Use DeleteSnapshot com um AWS SDK ou CLI](#)
- [Use DeleteSpotDatafeedSubscription com um CLI](#)
- [Use DeleteSubnet com um CLI](#)
- [Use DeleteTags com um CLI](#)
- [Use DeleteVolume com um CLI](#)
- [Use DeleteVpc com um AWS SDK ou CLI](#)

- [Use DeleteVpcEndpoint com um AWS SDK](#)
- [Use DeleteVpnConnection com um CLI](#)
- [Use DeleteVpnConnectionRoute com um CLI](#)
- [Use DeleteVpnGateway com um CLI](#)
- [Use DeregisterImage com um CLI](#)
- [Use DescribeAccountAttributes com um CLI](#)
- [Use DescribeAddresses com um AWS SDK ou CLI](#)
- [Use DescribeAvailabilityZones com um AWS SDK ou CLI](#)
- [Use DescribeBundleTasks com um CLI](#)
- [Use DescribeCapacityReservations com um CLI](#)
- [Use DescribeCustomerGateways com um CLI](#)
- [Use DescribeDhcpOptions com um CLI](#)
- [Use DescribeFlowLogs com um CLI](#)
- [Use DescribeHostReservationOfferings com um CLI](#)
- [Use DescribeHosts com um CLI](#)
- [Use DescribeIamInstanceProfileAssociations com um AWS SDK ou CLI](#)
- [Use DescribeIdFormat com um CLI](#)
- [Use DescribeIdentityIdFormat com um CLI](#)
- [Use DescribeImageAttribute com um CLI](#)
- [Use DescribeImages com um AWS SDK ou CLI](#)
- [Use DescribeImportImageTasks com um CLI](#)
- [Use DescribeImportSnapshotTasks com um CLI](#)
- [Use DescribeInstanceAttribute com um CLI](#)
- [Use DescribeInstanceStatus com um AWS SDK ou CLI](#)
- [Use DescribeInstanceTypes com um AWS SDK ou CLI](#)
- [Use DescribeInstances com um AWS SDK ou CLI](#)
- [Use DescribeInternetGateways com um CLI](#)
- [Use DescribeKeyPairs com um AWS SDK ou CLI](#)
- [Use DescribeNetworkAcls com um CLI](#)
- [Use DescribeNetworkInterfaceAttribute com um CLI](#)

- [Use DescribeNetworkInterfaces com um CLI](#)
- [Use DescribePlacementGroups com um CLI](#)
- [Use DescribePrefixLists com um CLI](#)
- [Use DescribeRegions com um AWS SDK ou CLI](#)
- [Use DescribeRouteTables com um AWS SDK ou CLI](#)
- [Use DescribeScheduledInstanceAvailability com um CLI](#)
- [Use DescribeScheduledInstances com um CLI](#)
- [Use DescribeSecurityGroups com um AWS SDK ou CLI](#)
- [Use DescribeSnapshotAttribute com um CLI](#)
- [Use DescribeSnapshots com um AWS SDK ou CLI](#)
- [Use DescribeSpotDatafeedSubscription com um CLI](#)
- [Use DescribeSpotFleetInstances com um CLI](#)
- [Use DescribeSpotFleetRequestHistory com um CLI](#)
- [Use DescribeSpotFleetRequests com um CLI](#)
- [Use DescribeSpotInstanceRequests com um CLI](#)
- [Use DescribeSpotPriceHistory com um CLI](#)
- [Use DescribeSubnets com um AWS SDK ou CLI](#)
- [Use DescribeTags com um CLI](#)
- [Use DescribeVolumeAttribute com um CLI](#)
- [Use DescribeVolumeStatus com um CLI](#)
- [Use DescribeVolumes com um CLI](#)
- [Use DescribeVpcAttribute com um CLI](#)
- [Use DescribeVpcClassicLink com um CLI](#)
- [Use DescribeVpcClassicLinkDnsSupport com um CLI](#)
- [Use DescribeVpcEndpointServices com um CLI](#)
- [Use DescribeVpcEndpoints com um CLI](#)
- [Use DescribeVpcs com um AWS SDK ou CLI](#)
- [Use DescribeVpnConnections com um CLI](#)
- [Use DescribeVpnGateways com um CLI](#)
- [Use DetachInternetGateway com um CLI](#)

- [Use DetachNetworkInterface com um CLI](#)
- [Use DetachVolume com um CLI](#)
- [Use DetachVpnGateway com um CLI](#)
- [Use DisableVgwRoutePropagation com um CLI](#)
- [Use DisableVpcClassicLink com um CLI](#)
- [Use DisableVpcClassicLinkDnsSupport com um CLI](#)
- [Use DisassociateAddress com um AWS SDK ou CLI](#)
- [Use DisassociateRouteTable com um CLI](#)
- [Use EnableVgwRoutePropagation com um CLI](#)
- [Use EnableVolumelo com um CLI](#)
- [Use EnableVpcClassicLink com um CLI](#)
- [Use EnableVpcClassicLinkDnsSupport com um CLI](#)
- [Use GetConsoleOutput com um CLI](#)
- [Use GetHostReservationPurchasePreview com um CLI](#)
- [Use GetPasswordData com um AWS SDK ou CLI](#)
- [Use ImportImage com um CLI](#)
- [Use ImportKeyPair com um CLI](#)
- [Use ImportSnapshot com um CLI](#)
- [Use ModifyCapacityReservation com um CLI](#)
- [Use ModifyHosts com um CLI](#)
- [Use ModifyIdFormat com um CLI](#)
- [Use ModifyImageAttribute com um CLI](#)
- [Use ModifyInstanceAttribute com um CLI](#)
- [Use ModifyInstanceCreditSpecification com um CLI](#)
- [Use ModifyNetworkInterfaceAttribute com um CLI](#)
- [Use ModifyReservedInstances com um CLI](#)
- [Use ModifySnapshotAttribute com um CLI](#)
- [Use ModifySpotFleetRequest com um CLI](#)
- [Use ModifySubnetAttribute com um CLI](#)
- [Use ModifyVolumeAttribute com um CLI](#)

- [Use ModifyVpcAttribute com um CLI](#)
- [Use MonitorInstances com um AWS SDK ou CLI](#)
- [Use MoveAddressToVpc com um CLI](#)
- [Use PurchaseHostReservation com um CLI](#)
- [Use PurchaseScheduledInstances com um CLI](#)
- [Use RebootInstances com um AWS SDK ou CLI](#)
- [Use RegisterImage com um CLI](#)
- [Use RejectVpcPeeringConnection com um CLI](#)
- [Use ReleaseAddress com um AWS SDK ou CLI](#)
- [Use ReleaseHosts com um CLI](#)
- [Use ReplacelamInstanceProfileAssociation com um AWS SDK ou CLI](#)
- [Use ReplaceNetworkAclAssociation com um CLI](#)
- [Use ReplaceNetworkAclEntry com um CLI](#)
- [Use ReplaceRoute com um CLI](#)
- [Use ReplaceRouteTableAssociation com um CLI](#)
- [Use ReportInstanceStatus com um CLI](#)
- [Use RequestSpotFleet com um CLI](#)
- [Use RequestSpotInstances com um CLI](#)
- [Use ResetImageAttribute com um CLI](#)
- [Use ResetInstanceAttribute com um CLI](#)
- [Use ResetNetworkInterfaceAttribute com um CLI](#)
- [Use ResetSnapshotAttribute com um CLI](#)
- [Use RevokeSecurityGroupEgress com um CLI](#)
- [Use RevokeSecurityGroupIngress com um CLI](#)
- [Use RunInstances com um AWS SDK ou CLI](#)
- [Use RunScheduledInstances com um CLI](#)
- [Use StartInstances com um AWS SDK ou CLI](#)
- [Use StopInstances com um AWS SDK ou CLI](#)
- [Use TerminateInstances com um AWS SDK ou CLI](#)
- [Use UnassignPrivateIpAddresses com um CLI](#)

- [Use UnmonitorInstances com um AWS SDK ou CLI](#)
- [Use UpdateSecurityGroupRuleDescriptionsIngress com um CLI](#)

Olá Amazon EC2

Os exemplos de código a seguir mostram como começar a usar a AmazonEC2.

.NET

AWS SDK for .NET

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
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>Async task.</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>();
    }
}
```

```
try
{
    // Retrieve information for up to 10 Amazon EC2 security groups.
    var request = new DescribeSecurityGroupsRequest { MaxResults = 10, };
    var securityGroups = new List<SecurityGroup>();

    var paginatorForSecurityGroups =
        ec2Client.Paginators.DescribeSecurityGroups(request);

    await foreach (var securityGroup in
paginatorForSecurityGroups.SecurityGroups)
    {
        securityGroups.Add(securityGroup);
    }

    // Now print the security groups returned by the call to
    // DescribeSecurityGroupsAsync.
    Console.WriteLine("Welcome to the EC2 Hello Service example. " +
        "\nLet's list your Security Groups:");
    securityGroups.ForEach(group =>
    {
        Console.WriteLine(
            $"Security group: {group.GroupName} ID: {group.GroupId}");
    });
}
catch (AmazonEC2Exception ex)
{
    Console.WriteLine($"An Amazon EC2 service error occurred while
listing security groups. {ex.Message}");
}
catch (Exception ex)
{
    Console.WriteLine($"An error occurred while listing security groups.
{ex.Message}");
}
}
```

- Para API obter detalhes, consulte [DescribeSecurityGroups](#) em AWS SDK for .NET API Referência.

C++

SDK para C++

 Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

Código para o CMakeLists arquivo.txtCMake.

```
# 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})
```

Código para o arquivo fonte hello_ec2.cpp.

```
#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) {
    (void)argc;
    (void)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) {
        Aws::EC2::Model::DescribeInstancesOutcome 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;
}

```

- Para API obter detalhes, consulte [DescribeSecurityGroup](#) em AWS SDK for C++ API Referência.

Java

SDK para Java 2.x

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
/**
 * Asynchronously describes the security groups for the specified group ID.
 *
 * @param groupName the name of the security group to describe
 * @return a {@link CompletableFuture} that represents the asynchronous
operation
 *         of describing the security groups. The future will complete with a
 *         {@link DescribeSecurityGroupsResponse} object that contains the
 *         security group information.
 */
public CompletableFuture<String> describeSecurityGroupArnByNameAsync(String
groupName) {
    DescribeSecurityGroupsRequest request =
DescribeSecurityGroupsRequest.builder()
        .groupNames(groupName)
        .build();

    DescribeSecurityGroupsPublisher paginator =
getAsyncClient().describeSecurityGroupsPaginator(request);
    AtomicReference<String> groupIdRef = new AtomicReference<>();
    return paginator.subscribe(response -> {
        response.securityGroups().stream()
            .filter(securityGroup ->
securityGroup.groupName().equals(groupName))
            .findFirst()
    });
}
```

```
        .ifPresent(securityGroup ->
groupIdRef.set(securityGroup.groupId()));
    }).thenApply(v -> {
        String groupId = groupIdRef.get();
        if (groupId == null) {
            throw new RuntimeException("No security group found with the
name: " + groupName);
        }
        return groupId;
    }).exceptionally(ex -> {
        logger.info("Failed to describe security group: " + ex.getMessage());
        throw new RuntimeException("Failed to describe security group", ex);
    });
}
```

- Para API obter detalhes, consulte [DescribeSecurityGroup](#) em AWS SDK for Java 2.x API Referência.

JavaScript

SDK para JavaScript (v3)

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
import { DescribeSecurityGroupsCommand, EC2Client } from "@aws-sdk/client-ec2";

// Call DescribeSecurityGroups and display the result.
export const main = async () => {
    const client = new EC2Client();
    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);
}
};

// Call function if run directly.
import { fileURLToPath } from "node:url";
if (process.argv[1] === fileURLToPath(import.meta.url)) {
  main();
}
```

- Para API obter detalhes, consulte [DescribeSecurityGroup](#) em AWS SDK for JavaScript API Referência.

Kotlin

SDK para Kotlin

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
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}")
}
}
```

- Para API obter detalhes, consulte a [DescribeSecurityGroups](#) referência AWS SDK do Kotlin API.

Python

SDK para Python (Boto3)

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
def hello_ec2(ec2_client):
    """
    Use the AWS SDK for Python (Boto3) to list the security groups in your
    account.
    This example uses the default settings specified in your shared credentials
    and config files.

    :param ec2_client: A Boto3 EC2 client. This client provides low-level
        access to AWS EC2 services.
    """
    print("Hello, Amazon EC2! Let's list up to 10 of your security groups:")
    try:
        paginator = ec2_client.get_paginator("describe_security_groups")
        response_iterator = paginator.paginate(MaxResults=10)
        for page in response_iterator:
            for sg in page["SecurityGroups"]:
                logger.info(f"\t{sg['GroupId']}: {sg['GroupName']}")
    except ClientError as err:
        logger.error("Failed to list security groups.")
        if err.response["Error"]["Code"] == "AccessDeniedException":
```

```
        logger.error("You do not have permission to list security groups.")
        raise

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

- Para API obter detalhes, consulte a [DescribeSecurityGroups](#) Referência AWS SDK do Python (Boto3). API

Ruby

SDK para Ruby

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
require 'aws-sdk-ec2'
require 'logger'

# EC2Manager is a class responsible for managing EC2 operations
# such as listing all EC2 instances in the current AWS account.
class EC2Manager
  def initialize(client)
    @client = client
    @logger = Logger.new($stdout)
  end

  # Lists and prints all EC2 instances in the current AWS account.
  def list_instances
    @logger.info('Listing instances')

    instances = fetch_instances

    if instances.empty?
      @logger.info('You have no instances')
```



```
    else
      print_instances(instances)
    end
  end

  private

  # Fetches all EC2 instances using pagination.
  #
  # @return [Array<Aws::EC2::Types::Instance>] List of EC2 instances.
  def fetch_instances
    paginator = @client.describe_instances
    instances = []

    paginator.each_page do |page|
      page.reservations.each do |reservation|
        reservation.instances.each do |instance|
          instances << instance
        end
      end
    end

    instances
  end

  # Prints details of the given EC2 instances.
  #
  # @param instances [Array<Aws::EC2::Types::Instance>] List of EC2 instances to
  # print.
  def print_instances(instances)
    instances.each do |instance|
      @logger.info("Instance ID: #{instance.instance_id}")
      @logger.info("Instance Type: #{instance.instance_type}")
      @logger.info("Public IP: #{instance.public_ip_address}")
      @logger.info("Public DNS Name: #{instance.public_dns_name}")
      @logger.info("\n")
    end
  end
end

if $PROGRAM_NAME == __FILE__
  ec2_client = Aws::EC2::Client.new(region: 'us-west-2')
  manager = EC2Manager.new(ec2_client)
  manager.list_instances
end
```

```
end
```

- Para API obter detalhes, consulte [DescribeSecurityGroup](#) em AWS SDK for Ruby API Referência.

Rust

SDK para Rust

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
async fn show_security_groups(client: &aws_sdk_ec2::Client, group_ids:
Vec<String>) {
    let response = client
        .describe_security_groups()
        .set_group_ids(Some(group_ids))
        .send()
        .await;

    match response {
        Ok(output) => {
            for group in output.security_groups() {
                println!(
                    "Found Security Group {} ({}), vpc id {} and description {}",
                    group.group_name().unwrap_or("unknown"),
                    group.group_id().unwrap_or("id-unknown"),
                    group.vpc_id().unwrap_or("vpcid-unknown"),
                    group.description().unwrap_or("(none)")
                );
            }
        }
        Err(err) => {
            let err = err.into_service_error();
            let meta = err.meta();
            let message = meta.message().unwrap_or("unknown");
        }
    }
}
```

```
        let code = meta.code().unwrap_or("unknown");
        eprintln!("Error listing EC2 Security Groups: ({code}) {message}");
    }
}
```

- Para API obter detalhes, consulte [DescribeSecurityGroups](#) a AWS SDK API referência do Rust.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Aprenda o básico da Amazon EC2 com um AWS SDK

Os exemplos de código a seguir mostram como:

- Criar um par de chaves e um grupo de segurança.
- Selecione uma Amazon Machine Image (AMI) e um tipo de instância compatível e, em seguida, crie uma instância.
- Interromper e reiniciar a instância.
- Associar um endereço IP elástico à sua instância.
- Conecte-se à sua instância eSSH, em seguida, limpe os recursos.

.NET

AWS SDK for .NET

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

Execute um cenário em um prompt de comando.

```
/// <summary>
```

```
/// Show Amazon Elastic Compute Cloud (Amazon EC2) Basics actions.
/// </summary>
public class EC2Basics
{
    public static ILogger<EC2Basics> _logger = null!;
    public static EC2Wrapper _ec2Wrapper = null!;
    public static SsmWrapper _ssmWrapper = null!;
    public static UiMethods _uiMethods = null!;

    public static string associationId = null!;
    public static string allocationId = null!;
    public static string instanceId = null!;
    public static string keyPairName = null!;
    public static string groupName = null!;
    public static string tempFileName = null!;
    public static string secGroupId = null!;
    public static bool isInteractive = true;

    /// <summary>
    /// Perform the actions defined for the Amazon EC2 Basics scenario.
    /// </summary>
    /// <param name="args">Command line arguments.</param>
    /// <returns>A Task object.</returns>
    public static async Task Main(string[] args)
    {
        // Set up dependency injection for Amazon EC2 and Amazon Simple Systems
        // Management (Amazon SSM) Service.
        using var host =
Microsoft.Extensions.Hosting.Host.CreateDefaultBuilder(args)
        .ConfigureServices((_, services) =>
            services.AddAWSService<IAmazonEC2>()
                .AddAWSService<IAmazonSimpleSystemsManagement>()
                .AddTransient<EC2Wrapper>()
                .AddTransient<SsmWrapper>()
        )
        .Build();

        SetUpServices(host);

        var uniqueName = Guid.NewGuid().ToString();
        keyPairName = "mvp-example-key-pair" + uniqueName;
        groupName = "ec2-scenario-group" + uniqueName;
        var groupDescription = "A security group created for the EC2 Basics
scenario.";
```

```
try
{
    // Start the scenario.
    _uiMethods.DisplayOverview();
    _uiMethods.PressEnter(isInteractive);

    // Create the key pair.
    _uiMethods.DisplayTitle("Create RSA key pair");
    Console.WriteLine("Let's create an RSA key pair that you can be use to
");
    Console.WriteLine("securely connect to your EC2 instance.");
    var keyPair = await _ec2Wrapper.CreateKeyPair(keyPairName);

    // Save key pair information to a temporary file.
    tempFileName = _ec2Wrapper.SaveKeyPair(keyPair);

    Console.WriteLine(
        $"Created the key pair: {keyPair.KeyName} and saved it to:
{tempFileName}");
    string? answer = "";
    if (isInteractive)
    {
        do
        {
            Console.WriteLine("Would you like to list your existing key
pairs? ");

            answer = Console.ReadLine();
        } while (answer!.ToLower() != "y" && answer.ToLower() != "n");
    }

    if (!isInteractive || 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 _ec2Wrapper.DescribeKeyPairs("");
        keyPairs.ForEach(kp =>
        {
            Console.WriteLine(
```

```
                $"{kp.KeyName} created at: {kp.CreateTime} Fingerprint:
{kp.KeyFingerprint}");
            });
        }

        _uiMethods.PressEnter(isInteractive);

        // Create the security group.
        Console.WriteLine(
            "Let's create a security group to manage access to your
instance.");
        secGroupId = await _ec2Wrapper.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 _ec2Wrapper.DescribeSecurityGroups(secGroupId);

        Console.WriteLine($"Created security group {groupName} in your
default VPC.");
        secGroups.ForEach(group =>
        {
            _ec2Wrapper.DisplaySecurityGroupInfoAsync(group);
        });
        _uiMethods.PressEnter(isInteractive);

        Console.WriteLine(
            "Now we'll authorize the security group we just created so that
it can");
        Console.WriteLine("access the EC2 instances you create.");
        await _ec2Wrapper.AuthorizeSecurityGroupIngress(groupName);

        secGroups = await _ec2Wrapper.DescribeSecurityGroups(secGroupId);
        Console.WriteLine($"Now let's look at the permissions again.");
        secGroups.ForEach(group =>
        {
            _ec2Wrapper.DisplaySecurityGroupInfoAsync(group);
        });
        _uiMethods.PressEnter(isInteractive);

        // Get list of available Amazon Linux 2 Amazon Machine Images (AMIs).
        var parameters =
```

```
        await _ssmWrapper.GetParametersByPath(
            "/aws/service/ami-amazon-linux-latest");

        List<string> imageIds = parameters.Select(param =>
param.Value).ToList();

        var images = await _ec2Wrapper.DescribeImages(imageIds);

        var i = 1;
        images.ForEach(image =>
        {
            Console.WriteLine($"{i++}\t{image.Description}");
        });

        int choice = 1;
        bool validNumber = false;
        if (isInteractive)
        {
            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
            _ec2Wrapper.DescribeInstanceTypes(selectedImage.Architecture);

        i = 1;
        instanceTypes.ForEach(instanceType =>
        {
            Console.WriteLine($"{i++}\t{instanceType.InstanceType}");
        });
        if (isInteractive)
        {
            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");
instanceId = await _ec2Wrapper.RunInstances(selectedImage.ImageId,
    selectedInstanceType, keyPairName, secGroupId);

_uiMethods.PressEnter(isInteractive);

var instance = await _ec2Wrapper.DescribeInstance(instanceId);
_uiMethods.DisplayTitle("New Instance Information");
_ec2Wrapper.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(isInteractive);

Console.WriteLine(
    "Now we'll stop the instance and then start it again to see
what's changed.");

await _ec2Wrapper.StopInstances(instanceId);

Console.WriteLine("Now let's start it up again.");
await _ec2Wrapper.StartInstances(instanceId);

Console.WriteLine("\nLet's see what changed.");

instance = await _ec2Wrapper.DescribeInstance(instanceId);
_uiMethods.DisplayTitle("New Instance Information");
_ec2Wrapper.DisplayInstanceInformation(instance);

Console.WriteLine("\nNotice the change in the SSH information:");
Console.WriteLine(
    $"{"\tssh -i {tempFileName} ec2-user@{instance.PublicIpAddress}"}");
```



```
        _uiMethods.PressEnter(isInteractive);

        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 _ec2Wrapper.StopInstances(instanceId);
        _uiMethods.PressEnter(isInteractive);

        _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 allocationResponse = await _ec2Wrapper.AllocateAddress();
        allocationId = allocationResponse.AllocationId;
        Console.WriteLine(
            "Now we will associate the Elastic IP address with our
instance.");
        associationId = await _ec2Wrapper.AssociateAddress(allocationId,
instanceId);

        // Start the instance again.
        Console.WriteLine("Now let's start the instance again.");
        await _ec2Wrapper.StartInstances(instanceId);

        Console.WriteLine("\nLet's see what changed.");

        instance = await _ec2Wrapper.DescribeInstance(instanceId);
        _uiMethods.DisplayTitle("Instance information");
        _ec2Wrapper.DisplayInstanceInformation(instance);

        Console.WriteLine("\nHere is the SSH information:");
        Console.WriteLine(
            $"{tempFileName} ec2-user@{instance.PublicIpAddress}");

        Console.WriteLine("Let's stop and start the instance again.");
        _uiMethods.PressEnter(isInteractive);

        await _ec2Wrapper.StopInstances(instanceId);

        Console.WriteLine("\nThe instance has stopped.");
```

```
        Console.WriteLine("Now let's start it up again.");
        await _ec2Wrapper.StartInstances(instanceId);

        instance = await _ec2Wrapper.DescribeInstance(instanceId);
        _uiMethods.DisplayTitle("New Instance Information");
        _ec2Wrapper.DisplayInstanceInformation(instance);
        Console.WriteLine("Note that the IP address did not change this
time.");
        _uiMethods.PressEnter(isInteractive);

        await Cleanup();
    }
    catch (Exception ex)
    {
        _logger.LogError(ex, "There was a problem with the scenario, starting
cleanup.");
        await Cleanup();
    }

    _uiMethods.DisplayTitle("EC2 Basics Scenario completed.");
    _uiMethods.PressEnter(isInteractive);
}

/// <summary>
/// Set up the services and logging.
/// </summary>
/// <param name="host"></param>
public static void SetUpServices(IHost host)
{
    var loggerFactory = LoggerFactory.Create(builder =>
    {
        builder.AddConsole();
    });
    _logger = new Logger<EC2Basics>(loggerFactory);

    // Now the client is available for injection.
    _ec2Wrapper = host.Services.GetRequiredService<EC2Wrapper>();
    _ssmWrapper = host.Services.GetRequiredService<SsmWrapper>();
    _uiMethods = new UiMethods();
}

/// <summary>
/// Clean up any resources from the scenario.
/// </summary>
```

```

    /// <returns></returns>
    public static async Task Cleanup()
    {
        _uiMethods.DisplayTitle("Clean up resources");
        Console.WriteLine("Now let's clean up the resources we created.");

        Console.WriteLine("Disassociate the Elastic IP address and release it.");
        // Disassociate the Elastic IP address.
        await _ec2Wrapper.DisassociateIp(associationId);

        // Delete the Elastic IP address.
        await _ec2Wrapper.ReleaseAddress(allocationId);

        // Terminate the instance.
        Console.WriteLine("Terminating the instance we created.");
        await _ec2Wrapper.TerminateInstances(instanceId);

        // Delete the security group.
        Console.WriteLine($"Deleting the Security Group: {groupName}.");
        await _ec2Wrapper.DeleteSecurityGroup(secGroupId);

        // Delete the RSA key pair.
        Console.WriteLine($"Deleting the key pair: {keyPairName}");
        await _ec2Wrapper.DeleteKeyPair(keyPairName);
        Console.WriteLine("Deleting the temporary file with the key
information.");
        _ec2Wrapper.DeleteTempFile(tempFileName);
        _uiMethods.PressEnter(isInteractive);
    }
}

```

Defina uma classe que envolva EC2 ações.

```

    /// <summary>
    /// Methods of this class perform Amazon Elastic Compute Cloud (Amazon EC2).
    /// </summary>
    public class EC2Wrapper
    {
        private readonly IAmazonEC2 _amazonEC2;
        private readonly ILogger<EC2Wrapper> _logger;

        /// <summary>

```

```
/// Constructor for the EC2Wrapper class.
/// </summary>
/// <param name="amazonScheduler">The injected EC2 client.</param>
/// <param name="logger">The injected logger.</param>
public EC2Wrapper(IAmazonEC2 amazonService, ILogger<EC2Wrapper> logger)
{
    _amazonEC2 = amazonService;
    _logger = logger;
}

/// <summary>
/// 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.
/// </summary>
/// <returns>The response object for the allocated address.</returns>
public async Task<AllocateAddressResponse> AllocateAddress()
{
    var request = new AllocateAddressRequest();

    try
    {
        var response = await _amazonEC2.AllocateAddressAsync(request);
        Console.WriteLine($"Allocated IP: {response.PublicIp} with allocation
ID {response.AllocationId}.");
        return response;
    }
    catch (AmazonEC2Exception ec2Exception)
    {
        if (ec2Exception.ErrorCode == "AddressLimitExceeded")
        {
            // For more information on Elastic IP address quotas, see:
            // https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/elastic-
ip-addresses-eip.html#using-instance-addressing-limit
            _logger.LogError($"Unable to allocate Elastic IP, address limit
exceeded. {ec2Exception.Message}");
        }

        throw;
    }
    catch (Exception ex)
    {

```

```
        _logger.LogError($"An error occurred while allocating Elastic IP.:  
{ex.Message}");  
        throw;  
    }  
}  
  
/// <summary>  
/// 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.  
/// </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)  
{  
    try  
    {  
        var request = new AssociateAddressRequest  
        {  
            AllocationId = allocationId,  
            InstanceId = instanceId  
        };  
  
        var response = await _amazonEC2.AssociateAddressAsync(request);  
        return response.AssociationId;  
    }  
    catch (AmazonEC2Exception ec2Exception)  
    {  
        if (ec2Exception.ErrorCode == "InvalidInstanceId")  
        {  
            _logger.LogError(  
                $"InstanceId is invalid, unable to associate address.  
{ec2Exception.Message}");  
        }  
  
        throw;  
    }  
}
```

```
        catch (Exception ex)
        {
            _logger.LogError(
                $"An error occurred while associating the Elastic IP.:
{ex.Message}");
            throw;
        }
    }

    /// <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)
    {
        try
        {
            // 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;
        }
        catch (AmazonEC2Exception ec2Exception)
        {
            if (ec2Exception.ErrorCode == "InvalidPermission.Duplicate")
            {
                _logger.LogError(
                    $"The ingress rule already exists. {ec2Exception.Message}");
            }
        }
    }
}
```

```
        throw;
    }
    catch (Exception ex)
    {
        _logger.LogError(
            $"An error occurred while authorizing ingress.: {ex.Message}");
        throw;
    }
}

/// <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 with a specified name.
/// </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)
{
    try
    {
        var request = new CreateKeyPairRequest { KeyName = keyPairName, };

        var response = await _amazonEC2.CreateKeyPairAsync(request);

        var kp = response.KeyPair;
        // Return the key pair so it can be saved if needed.
    }
}
```

```
// Wait until the key pair exists.
int retries = 5;
while (retries-- > 0)
{
    Console.WriteLine($"Checking for new KeyPair {keyPairName}...");
    var keyPairs = await DescribeKeyPairs(keyPairName);
    if (keyPairs.Any())
    {
        return kp;
    }

    Thread.Sleep(5000);
    retries--;
}
_logger.LogError($"Unable to find newly created KeyPair
{keyPairName}.");
throw new DoesNotExistException("KeyPair not found");
}
catch (AmazonEC2Exception ec2Exception)
{
    if (ec2Exception.ErrorCode == "InvalidKeyPair.Duplicate")
    {
        _logger.LogError(
            $"A key pair called {keyPairName} already exists.");
    }

    throw;
}
catch (Exception ex)
{
    _logger.LogError(
        $"An error occurred while creating the key pair.: {ex.Message}");
    throw;
}
}

/// <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 with a specified name and
description.
/// </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)
{
    try
    {
        var response = await _amazonEC2.CreateSecurityGroupAsync(
            new CreateSecurityGroupRequest(groupName, groupDescription));

        // Wait until the security group exists.
        int retries = 5;
        while (retries-- > 0)
        {
            var groups = await DescribeSecurityGroups(response.GroupId);
            if (groups.Any())
            {
                return response.GroupId;
            }

            Thread.Sleep(5000);
            retries--;
        }
        _logger.LogError($"Unable to find newly created group {groupName}.");
        throw new DoesNotExistException("security group not found");
    }
    catch (AmazonEC2Exception ec2Exception)
```

```
        {
            if (ec2Exception.ErrorCode == "ResourceAlreadyExists")
            {
                _logger.LogError(
                    $"A security group with the name {groupName} already exists.
{ec2Exception.Message}");
            }
            throw;
        }
        catch (Exception ex)
        {
            _logger.LogError(
                $"An error occurred while creating the security group.:
{ex.Message}");
            throw;
        }
    }

    /// <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 (AmazonEC2Exception ec2Exception)
    {
        if (ec2Exception.ErrorCode == "InvalidKeyPair.NotFound")
        {
            _logger.LogError($"KeyPair {keyPairName} does not exist and
cannot be deleted. Please verify the key pair name and try again.");
        }

        return false;
    }
    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)
{
    try
    {
        var response =
            await _amazonEC2.DeleteSecurityGroupAsync(
                new DeleteSecurityGroupRequest { GroupId = groupId });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }
    catch (AmazonEC2Exception ec2Exception)
    {
        if (ec2Exception.ErrorCode == "InvalidGroup.NotFound")
        {
            _logger.LogError(
                $"Security Group {groupId} does not exist and cannot be
deleted. Please verify the ID and try again.");
        }

        return false;
    }
    catch (Exception ex)
    {
        Console.WriteLine($"Couldn't delete the security group because:
{ex.Message}");
        return false;
    }
}

/// <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 EC2 instances with a particular state.
    /// </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>True if successful.</returns>
    public async Task<bool> GetInstancesWithState(string state)
    {
        try
        {
            // Filters the results of the instance list.
            var filters = new List<Filter>
            {
                new Filter
                {
                    Name = $"instance-state-name",
                    Values = new List<string> { state, },
                },
            };
            var request = new DescribeInstancesRequest { Filters = filters, };

            Console.WriteLine($"\\nShowing instances with state {state}");
            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.WriteLine($"Instance ID: {instance.InstanceId} ");
                    Console.WriteLine($"\\tCurrent State:
{instance.State.Name}");
                }
            }
        }

        return true;
    }
    catch (AmazonEC2Exception ec2Exception)
    {
        if (ec2Exception.ErrorCode == "InvalidParameterValue")
        {
            _logger.LogError(
                $"Invalid parameter value for filtering instances.");
        }

        return false;
    }
    catch (Exception ex)
    {
        Console.WriteLine($"Couldn't list instances because: {ex.Message}");
        return false;
    }
}

/// <summary>
/// Describe the instance types available.
/// </summary>
/// <returns>A list of instance type information.</returns>
public async Task<List<InstanceTypeInfo>>
DescribeInstanceTypes(ArchitectureValues architecture)
{
    try
    {
        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;
    }
    catch (AmazonEC2Exception ec2Exception)
    {
        if (ec2Exception.ErrorCode == "InvalidParameterValue")
        {
            _logger.LogError(
                $"Parameters are invalid. Ensure architecture and size
strings conform to DescribeInstanceTypes API reference.");
        }

        throw;
    }
    catch (Exception ex)
    {
        Console.WriteLine($"Couldn't delete the security group because:
{ex.Message}");
        throw;
    }
}

/// <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)
```



```
{
    try
    {
        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();
    }
    catch (AmazonEC2Exception ec2Exception)
    {
        if (ec2Exception.ErrorCode == "InvalidKeyPair.NotFound")
        {
            _logger.LogError(
                $"A key pair called {keyPairName} does not exist.");
        }

        throw;
    }
    catch (Exception ex)
    {
        _logger.LogError(
            $"An error occurred while describing the key pair.:
{ex.Message}");
        throw;
    }
}

/// <summary>
/// Retrieve information for one or all Amazon EC2 security group.
/// </summary>
/// <param name="groupId">The optional Id of a specific Amazon EC2 security
group.</param>
/// <returns>A list of security group information.</returns>
public async Task<List<SecurityGroup>> DescribeSecurityGroups(string groupId)
{
    try
```

```
{
    var securityGroups = new List<SecurityGroup>();
    var request = new DescribeSecurityGroupsRequest();

    if (!string.IsNullOrEmpty(groupId))
    {
        var groupIds = new List<string> { groupId };
        request.GroupIds = groupIds;
    }

    var paginatorForSecurityGroups =
        _amazonEC2.Paginators.DescribeSecurityGroups(request);

    await foreach (var securityGroup in
paginatorForSecurityGroups.SecurityGroups)
    {
        securityGroups.Add(securityGroup);
    }

    return securityGroups;
}
catch (AmazonEC2Exception ec2Exception)
{
    if (ec2Exception.ErrorCode == "InvalidGroup.NotFound")
    {
        _logger.LogError(
            $"A security group {groupId} does not exist.");
    }

    throw;
}
catch (Exception ex)
{
    _logger.LogError(
        $"An error occurred while listing security groups.
{ex.Message}");
    throw;
}
}

/// <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.WriteLine($"  \tIpv4Ranges: ");
        permission.Ipv4Ranges.ForEach(range =>
        { Console.WriteLine($"  \t{range.CidrIp} "); });

        Console.WriteLine($"  \n\tIpv6Ranges:");
        permission.Ipv6Ranges.ForEach(range =>
        { Console.WriteLine($"  \t{range.CidrIpv6} "); });

        Console.WriteLine($"  \n\tPrefixListIds: ");
        permission.PrefixListIds.ForEach(id => Console.WriteLine($"  \t{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.WriteLine($"  \tIpv4Ranges: ");
        permission.Ipv4Ranges.ForEach(range =>
        { Console.WriteLine($"  \t{range.CidrIp} "); });

        Console.WriteLine($"  \n\tIpv6Ranges:");
        permission.Ipv6Ranges.ForEach(range =>
        { Console.WriteLine($"  \t{range.CidrIpv6} "); });

        Console.WriteLine($"  \n\tPrefixListIds: ");
        permission.PrefixListIds.ForEach(id => Console.WriteLine($"  \t{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)
{
    try
    {
        var response = await _amazonEC2.DisassociateAddressAsync(
            new DisassociateAddressRequest { AssociationId =
associationId });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }
    catch (AmazonEC2Exception ec2Exception)
    {
        if (ec2Exception.ErrorCode == "InvalidAssociationID.NotFound")
        {
            _logger.LogError(
                $"AssociationId is invalid, unable to disassociate address.
{ec2Exception.Message}");
        }

        return false;
    }
    catch (Exception ex)
    {
        _logger.LogError(
            $"An error occurred while disassociating the Elastic IP.:
{ex.Message}");
        return false;
    }
}

/// <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 a specific EC2 instance.
/// </summary>
/// <param name="ec2InstanceId">The instance Id of the instance that will be
rebooted.</param>
/// <returns>Async Task.</returns>
public async Task<bool> RebootInstances(string ec2InstanceId)
{
    try
    {
        var request = new RebootInstancesRequest
        {
            InstanceIds = new List<string> { ec2InstanceId },
        };

        await _amazonEC2.RebootInstancesAsync(request);

        // Wait for the instance to be running.
        Console.WriteLine("Waiting for the instance to start.");
        await WaitForInstanceState(ec2InstanceId, InstanceStateName.Running);

        return true;
    }
    catch (AmazonEC2Exception ec2Exception)
    {
        if (ec2Exception.ErrorCode == "InvalidInstanceId")
        {
            _logger.LogError(
                $"InstanceId {ec2InstanceId} is invalid, unable to reboot.
{ec2Exception.Message}");
        }
        return false;
    }
    catch (Exception ex)
    {
        _logger.LogError(
            $"An error occurred while rebooting the instance
{ec2InstanceId}.: {ex.Message}");
    }
}
```

```
        return false;
    }
}

/// <summary>
/// Release an Elastic IP address. After the Elastic IP address is released,
/// it can no longer be used.
/// </summary>
/// <param name="allocationId">The allocation Id of the Elastic IP address.</
param>
/// <returns>True if successful.</returns>
public async Task<bool> ReleaseAddress(string allocationId)
{
    try
    {
        var request = new ReleaseAddressRequest { AllocationId =
allocationId };

        var response = await _amazonEC2.ReleaseAddressAsync(request);
        return response.HttpStatusCode == HttpStatusCode.OK;
    }
    catch (AmazonEC2Exception ec2Exception)
    {
        if (ec2Exception.ErrorCode == "InvalidAllocationID.NotFound")
        {
            _logger.LogError(
                $"AllocationId {allocationId} was not found.
{ec2Exception.Message}");
        }

        return false;
    }
    catch (Exception ex)
    {
        _logger.LogError(
            $"An error occurred while releasing the AllocationId
{allocationId}.: {ex.Message}");
        return false;
    }
}

/// <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)
    {
        try
        {
            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);
            var instanceId = response.Reservation.Instances[0].InstanceId;

            Console.WriteLine("Waiting for the instance to start.");
            await WaitForInstanceState(instanceId, InstanceStateName.Running);

            return instanceId;
        }
        catch (AmazonEC2Exception ec2Exception)
        {
            if (ec2Exception.ErrorCode == "InvalidGroupId.NotFound")
            {
                _logger.LogError(
                    $"GroupId {groupId} was not found. {ec2Exception.Message}");
            }

            throw;
        }
        catch (Exception ex)
        {
```

```
        _logger.LogError(
            $"An error occurred while running the instance.: {ex.Message}");
        throw;
    }
}

/// <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)
{
    try
    {
        var request = new StartInstancesRequest
        {
            InstanceIds = new List<string> { ec2InstanceId },
        };

        await _amazonEC2.StartInstancesAsync(request);

        Console.WriteLine("Waiting for instance to start. ");
        await WaitForInstanceState(ec2InstanceId, InstanceStateName.Running);
    }
    catch (AmazonEC2Exception ec2Exception)
    {
        if (ec2Exception.ErrorCode == "InvalidInstanceId")
        {
            _logger.LogError(
                $"InstanceId is invalid, unable to start.
{ec2Exception.Message}");
        }

        throw;
    }
    catch (Exception ex)
    {
        _logger.LogError(
            $"An error occurred while starting the instance.: {ex.Message}");
        throw;
    }
}
```



```
}

/// <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)
{
    try
    {
        var request = new StopInstancesRequest
        {
            InstanceIds = new List<string> { ec2InstanceId },
        };

        await _amazonEC2.StopInstancesAsync(request);
        Console.WriteLine("Waiting for the instance to stop.");
        await WaitForInstanceState(ec2InstanceId, InstanceStateName.Stopped);

        Console.WriteLine("\nThe instance has stopped.");
    }
    catch (AmazonEC2Exception ec2Exception)
    {
        if (ec2Exception.ErrorCode == "InvalidInstanceId")
        {
            _logger.LogError(
                $"InstanceId is invalid, unable to stop.
{ec2Exception.Message}");
        }

        throw;
    }
    catch (Exception ex)
    {
        _logger.LogError(
            $"An error occurred while stopping the instance.: {ex.Message}");
        throw;
    }
}

/// <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)
{
    try
    {
        var request = new TerminateInstancesRequest
        {
            InstanceIds = new List<string> { ec2InstanceId }
        };

        var response = await _amazonEC2.TerminateInstancesAsync(request);
        Console.WriteLine("Waiting for the instance to terminate.");
        await WaitForInstanceState(ec2InstanceId,
InstanceStateName.Terminated);

        Console.WriteLine($"\\nThe instance {ec2InstanceId} has been
terminated.");
        return response.TerminatingInstances;
    }
    catch (AmazonEC2Exception ec2Exception)
    {
        if (ec2Exception.ErrorCode == "InvalidInstanceId")
        {
            _logger.LogError(
                $"InstanceId is invalid, unable to terminate.
{ec2Exception.Message}");
        }

        throw;
    }
    catch (Exception ex)
    {
        _logger.LogError(
            $"An error occurred while terminating the instance.:
{ex.Message}");
        throw;
    }
}

/// <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 in the specified state.
    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.WriteLine(". ");
    } while (!hasState);

    return hasState;
}
}
```

- Para API obter detalhes, consulte os tópicos a seguir em AWS SDK for .NET APIReferência.
 - [AllocateAddress](#)
 - [AssociateAddress](#)
 - [AuthorizeSecurityGroupIngress](#)
 - [CreateKeyPair](#)
 - [CreateSecurityGroup](#)
 - [DeleteKeyPair](#)

- [DeleteSecurityGroup](#)
- [DescribeImages](#)
- [DescribeInstanceTypes](#)
- [DescribeInstances](#)
- [DescribeKeyPairs](#)
- [DescribeSecurityGroups](#)
- [DisassociateAddress](#)
- [ReleaseAddress](#)
- [RunInstances](#)
- [StartInstances](#)
- [StopInstances](#)
- [TerminateInstances](#)
- [UnmonitorInstances](#)

Bash

AWS CLI com script Bash

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

Execute um cenário interativo em um prompt de comando.

```
#####  
# 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 getopt 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 getopt "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
}

```

```

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
    fi
  done
}
```

```

    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
}
}
```

As funções do DynamoDB usadas nesse cenário.

```
#####
# 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 getopt 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 getopt 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 getopt "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 getopt 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 getopt "g:h" option; do
        case "${option}" in
            g) security_group_id="${OPTARG}" ;;
            h)
                usage
                return 0
                ;;
            \?)
                echo "Invalid parameter"
                usage
                return 1
        esac
    done
}

```

```

        ;;
    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 getopt 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 getopt "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
}
```

```

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 getopt 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 getopt "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 getopt 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 getopt "i:t:k:s:c:h" option; do
        case "${option}" in
            i) image_id="${OPTARG}" ;;
            t) instance_type="${OPTARG}" ;;
            k) key_pair_name="${OPTARG}" ;;
        esac
    done
}

```

```
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 getopt 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 getopt 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 getopt "i:h" option; do
        case "${option}" in
            i) instance_ids="${OPTARG}" ;;
            h)
                usage
                return 0
                ;;
        esac
    done
}
```

```

    \?)
    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 getopt 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
}
```

```

        ;;
    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
}
```

```

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 getopt 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 getopt "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 getopt 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 getopt "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 getopt 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 getopt "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
}

```

As funções utilitárias usadas nesse cenário.

```

#####
# 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
}
```

- Para API obter detalhes, consulte os tópicos a seguir na Referência de AWS CLI Comandos.
 - [AllocateAddress](#)
 - [AssociateAddress](#)
 - [AuthorizeSecurityGroupIngress](#)
 - [CreateKeyPair](#)
 - [CreateSecurityGroup](#)
 - [DeleteKeyPair](#)
 - [DeleteSecurityGroup](#)
 - [DescribeImages](#)
 - [DescribeInstanceTypes](#)
 - [DescribeInstances](#)
 - [DescribeKeyPairs](#)
 - [DescribeSecurityGroups](#)
 - [DisassociateAddress](#)
 - [ReleaseAddress](#)

- [RunInstances](#)
- [StartInstances](#)
- [StopInstances](#)
- [TerminateInstances](#)
- [UnmonitorInstances](#)

Java

SDK para Java 2.x

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

Execute um cenário em um prompt de comando.

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.services.ec2.model.CreateKeyPairResponse;
import software.amazon.awssdk.services.ec2.model.DeleteKeyPairResponse;
import software.amazon.awssdk.services.ec2.model.DescribeKeyPairsResponse;
import software.amazon.awssdk.services.ec2.model.DisassociateAddressResponse;
import software.amazon.awssdk.services.ec2.model.Ec2Exception;
import software.amazon.awssdk.services.ec2.model.ReleaseAddressResponse;
import software.amazon.awssdk.services.ssm.model.GetParametersByPathResponse;
import software.amazon.awssdk.services.ssm.model.Parameter;

import java.net.InetAddress;
import java.net.UnknownHostException;
import java.util.List;
import java.util.Scanner;
import java.util.concurrent.CompletableFuture;
import java.util.concurrent.CompletionException;

/**
 * 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 additional 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",
"-");
    private static final Logger logger =
LoggerFactory.getLogger(EC2Scenario.class);
    public static void main(String[] args) throws InterruptedException,
UnknownHostException {

        logger.info("""
Usage:
    <keyName> <fileName> <groupName> <groupDesc>

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
        """);

Scanner scanner = new Scanner(System.in);
EC2Actions ec2Actions = new EC2Actions();

String keyName = "TestKeyPair7" ;
String fileName = "ec2Key.pem";
String groupName = "TestSecGroup7" ;
String groupDesc = "Test Group" ;
String vpcId = ec2Actions.describeFirstEC2VpcAsync().join().vpcId();
InetAddress localAddress = InetAddress.getLocalHost();
String myIpAddress = localAddress.getHostAddress();

logger.info("""
    Amazon Elastic Compute Cloud (EC2) is a web service that provides
secure, resizable compute
    capacity in the cloud. It allows developers and organizations to
easily launch and manage
    virtual server instances, known as EC2 instances, to run their
applications.

    EC2 provides a wide range of instance types, each with different
compute, memory,
    and storage capabilities, to meet the diverse needs of various
workloads. Developers
    can choose the appropriate instance type based on their application's
requirements,
    such as high-performance computing, memory-intensive tasks, or GPU-
accelerated workloads.

    The `Ec2AsyncClient` interface in the AWS SDK for Java 2.x provides a
set of methods to
    programmatically interact with the Amazon EC2 service. This allows
developers to
    automate the provisioning, management, and monitoring of EC2
instances as part of their
    application deployment pipelines. With EC2, teams can focus on
building and deploying
    their applications without having to worry about the underlying
infrastructure
    required to host and manage physical servers.
```



```
    This scenario walks you through how to perform key operations for
    this service.
    Let's get started...
    """);

    waitForInputToContinue(scanner);
    logger.info(DASHES);

    logger.info(DASHES);
    logger.info("1. Create an RSA key pair and save the private key material
    as a .pem file.");
    logger.info("""
        An RSA key pair for Amazon EC2 is a security mechanism used to
        authenticate and secure
        access to your EC2 instances. It consists of a public key and a
        private key,
        which are generated as a pair.
    """);
    waitForInputToContinue(scanner);
    try {
        CompletableFuture<CreateKeyPairResponse> future =
    ec2Actions.createKeyPairAsync(keyName, fileName);
        CreateKeyPairResponse response = future.join();
        logger.info("Key Pair successfully created. Key Fingerprint: " +
    response.keyFingerprint());

    } catch (RuntimeException rt) {
        Throwable cause = rt.getCause();
        if (cause instanceof Ec2Exception ec2Ex) {
            if (ec2Ex.getMessage().contains("already exists")) {
                // Key pair already exists.
                logger.info("The key pair '" + keyName + "' already exists.
    Moving on...");
            } else {
                logger.info("EC2 error occurred: Error message: {}, Error
    code {}", ec2Ex.getMessage(), ec2Ex.awsErrorDetails().errorCode());
                return;
            }
        } else {
            logger.info("An unexpected error occurred: " +
    (rt.getMessage()));
            return;
        }
    }
}
```

```

    waitForInputToContinue(scanner);
    logger.info(DASHES);

    logger.info(DASHES);
    logger.info("2. List key pairs.");
    waitForInputToContinue(scanner);
    try {
        CompletableFuture<DescribeKeyPairsResponse> future =
ec2Actions.describeKeysAsync();
        DescribeKeyPairsResponse keyPairsResponse = future.join();
        keyPairsResponse.keyPairs().forEach(keyPair -> logger.info(
            "Found key pair with name {} and fingerprint {}",
            keyPair.keyName(),
            keyPair.keyFingerprint()));
    } catch (RuntimeException rt) {
        Throwable cause = rt.getCause();
        if (cause instanceof Ec2Exception ec2Ex) {
            logger.info("EC2 error occurred: Error message: {}, Error code
{}", ec2Ex.getMessage(), ec2Ex.awsErrorDetails().errorCode());
            return;
        } else {
            logger.info("An unexpected error occurred: {}", (cause != null ?
cause.getMessage() : rt.getMessage()));
            return;
        }
    }
    waitForInputToContinue(scanner);
    logger.info(DASHES);

    logger.info(DASHES);
    logger.info("3. Create a security group.");
    logger.info("""
        An AWS EC2 Security Group is a virtual firewall that controls the
inbound and outbound traffic to an EC2 instance. It acts as a first
line
of defense for your EC2 instances, allowing you to specify the rules
that
govern the network traffic entering and leaving your instances.
        """);
    waitForInputToContinue(scanner);
    String groupId = "";
    try {

```

```
        CompletableFuture<String> future =
ec2Actions.createSecurityGroupAsync(groupName, groupDesc, vpcId, myIpAddress);
        future.join();
        logger.info("Created security group" ) ;

    } catch (RuntimeException rt) {
        Throwable cause = rt.getCause();
        if (cause instanceof Ec2Exception ec2Ex) {
            if (ec2Ex.awsErrorDetails().errorMessage().contains("already
exists")) {
                logger.info("The Security Group already exists. Moving
on...");
            } else {
                logger.error("An unexpected error occurred: {}",
ec2Ex.awsErrorDetails().errorMessage());
                return;
            }
        } else {
            logger.error("An unexpected error occurred: {}",
cause.getMessage());
            return;
        }
    }
    waitForInputToContinue(scanner);
    logger.info(DASHES);

    logger.info(DASHES);
    logger.info("4. Display security group information for the new security
group.");
    waitForInputToContinue(scanner);
    try {
        CompletableFuture<String> future =
ec2Actions.describeSecurityGroupArnByNameAsync(groupName);
        groupId = future.join();
        logger.info("The security group Id is "+groupId);

    } catch (RuntimeException rt) {
        Throwable cause = rt.getCause();
        if (cause instanceof Ec2Exception ec2Ex) {
            String errorCode = ec2Ex.awsErrorDetails().errorCode();
            if ("InvalidGroup.NotFound".equals(errorCode)) {
                logger.info("Security group '{}' does not exist. Error Code:
{}", groupName, errorCode);
            } else {
```

```

        logger.info("EC2 error occurred: Message {}, Error Code: {}",
ec2Ex.getMessage(), errorCode);
    }
    } else {
        logger.info("An unexpected error occurred: {}",
cause.getMessage());
    }
}
waitForInputToContinue(scanner);
logger.info(DASHES);

logger.info(DASHES);
logger.info("5. Get a list of Amazon Linux 2 AMIs and select one with
amzn2 in the name.");
logger.info("""
    An Amazon EC2 AMI (Amazon Machine Image) is a pre-configured virtual
machine image that
    serves as a template for launching EC2 instances. It contains all the
necessary software and
    configurations required to run an application or operating system on
an EC2 instance.
    """);
waitForInputToContinue(scanner);
String instanceAMI="";
try {
    CompletableFuture<GetParametersByPathResponse> future =
ec2Actions.getParaValuesAsync();
    GetParametersByPathResponse pathResponse = future.join();
    List<Parameter> parameterList = pathResponse.parameters();
    for (Parameter para : parameterList) {
        if (filterName(para.name())) {
            instanceAMI = para.value();
            break;
        }
    }
} catch (RuntimeException rt) {
    Throwable cause = rt.getCause();
    if (cause instanceof Ec2Exception ec2Ex) {
        logger.info("EC2 error occurred: Message {}, Error Code:{}",
ec2Ex.getMessage(), ec2Ex.awsErrorDetails().errorCode());
        return;
    } else {
        logger.info("An unexpected error occurred: {}",
cause.getMessage());

```

```
        return;
    }
}
logger.info("The AMI value with amzn2 is: {}", instanceAMI);
waitForInputToContinue(scanner);
logger.info(DASHES);

logger.info(DASHES);
logger.info("6. Get the (Amazon Machine Image) AMI value from the amzn2
image.");
logger.info("""
    An AMI value represents a specific version of a virtual machine (VM)
or server image.
    It uniquely identifies a particular version of an EC2 instance,
including its operating system,
    pre-installed software, and any custom configurations. This allows you
to consistently deploy the same
    VM image across your infrastructure.

    """);
waitForInputToContinue(scanner);
String amiValue;
try {
    CompletableFuture<String> future =
ec2Actions.describeImageAsync(instanceAMI);
    amiValue = future.join();

} catch (CompletionException ce) {
    Throwable cause = ce.getCause();
    if (cause instanceof Ec2Exception) {
        Ec2Exception ec2Ex = (Ec2Exception) cause;
        logger.info("EC2 error occurred: Message {}, Error Code:{}",
ec2Ex.getMessage(), ec2Ex.awsErrorDetails().errorCode());
        return;
    } else {
        logger.info("An unexpected error occurred: {}",
cause.getMessage());
        return;
    }
}
waitForInputToContinue(scanner);
logger.info(DASHES);

logger.info(DASHES);
```

```
        logger.info("7. Retrieves an instance type available in the current AWS
region.");
        waitForInputToContinue(scanner);
        String instanceType;
        try {
            CompletableFuture<String> future =
ec2Actions.getInstanceTypesAsync();
            instanceType = future.join();
            if (!instanceType.isEmpty()) {
                logger.info("Found instance type: " + instanceType);
            } else {
                logger.info("Desired instance type not found.");
            }
        } catch (RuntimeException rt) {
            Throwable cause = rt.getCause();
            if (cause instanceof Ec2Exception ec2Ex) {
                logger.info("EC2 error occurred: Message {}, Error Code:{}",
ec2Ex.getMessage(), ec2Ex.awsErrorDetails().errorCode());
                return;
            } else {
                logger.info("An unexpected error occurred: {}",
cause.getMessage());
                return;
            }
        }
        waitForInputToContinue(scanner);
        logger.info(DASHES);

        logger.info(DASHES);
        logger.info("8. Create an Amazon EC2 instance using the key pair, the
instance type, the security group, and the EC2 AMI value.");
        logger.info("Once the EC2 instance is created, it is placed into a
running state.");
        waitForInputToContinue(scanner);
        String newInstanceId;
        try {
            CompletableFuture<String> future =
ec2Actions.runInstanceAsync(instanceType, keyName, groupName, amiValue);
            newInstanceId = future.join();
        } catch (RuntimeException rt) {
            Throwable cause = rt.getCause();
            if (cause instanceof Ec2Exception) {
                Ec2Exception ec2Ex = (Ec2Exception) cause;
                switch (ec2Ex.awsErrorDetails().errorCode()) {
```

```

        case "InvalidParameterValue":
            logger.info("EC2 error occurred: Message {}, Error Code:
{}", ec2Ex.getMessage(), ec2Ex.awsErrorDetails().errorCode());
            break;
        case "InsufficientInstanceCapacity":
            // Handle insufficient instance capacity.
            logger.info("Insufficient instance capacity: {}, {}",
ec2Ex.getMessage(), ec2Ex.awsErrorDetails().errorCode());
            break;
        case "InvalidGroup.NotFound":
            // Handle security group not found.
            logger.info("Security group not found: {},{}",
ec2Ex.getMessage(), ec2Ex.awsErrorDetails().errorCode());
            break;
        default:
            logger.info("EC2 error occurred: {} (Code: {})",
ec2Ex.getMessage(), ec2Ex.awsErrorDetails().errorCode());
            break;
    }
    return;
} else {
    logger.info("An unexpected error occurred: {}", (cause != null ?
cause.getMessage() : rt.getMessage()));
    return;
}
}
logger.info("The instance Id is " + newInstanceId);
waitForInputToContinue(scanner);
logger.info(DASHES);

logger.info(DASHES);
logger.info("9. Display information about the running instance. ");

waitForInputToContinue(scanner);
String publicIp;
try {
    CompletableFuture<String> future =
ec2Actions.describeEC2InstancesAsync(newInstanceId);
    publicIp = future.join();
    logger.info("EC2 instance public IP {}", publicIp);
} catch (RuntimeException rt) {
    Throwable cause = rt.getCause();
    if (cause instanceof Ec2Exception ec2Ex) {

```

```
        logger.info("EC2 error occurred: Message {}, Error Code:{}",
ec2Ex.getMessage(), ec2Ex.awsErrorDetails().errorCode());
        return;
    } else {
        logger.info("An unexpected error occurred: {}",
cause.getMessage());
        return;
    }

}
logger.info("You can SSH to the instance using this command:");
logger.info("ssh -i " + fileName + " ec2-user@" + publicIp);
waitForInputToContinue(scanner);
logger.info(DASHES);

logger.info(DASHES);
logger.info("10. Stop the instance using a waiter (this may take a few
mins).");
// Remove the 2nd one
waitForInputToContinue(scanner);
try {
    CompletableFuture<Void> future =
ec2Actions.stopInstanceAsync(newInstanceId);
    future.join();

} catch (RuntimeException rt) {
    Throwable cause = rt.getCause();
    if (cause instanceof Ec2Exception ec2Ex) {
        logger.info("EC2 error occurred: Message {}, Error Code:{}",
ec2Ex.getMessage(), ec2Ex.awsErrorDetails().errorCode());
        return;
    } else {
        logger.info("An unexpected error occurred: {}",
cause.getMessage());
        return;
    }
}
waitForInputToContinue(scanner);
logger.info(DASHES);

logger.info(DASHES);
logger.info("11. Start the instance using a waiter (this may take a few
mins).");
try {
```



```
        CompletableFuture<Void> future =
ec2Actions.startInstanceAsync(newInstanceId);
        future.join();

    } catch (RuntimeException rt) {
        Throwable cause = rt.getCause();
        if (cause instanceof Ec2Exception ec2Ex) {
            // Handle EC2 exceptions.
            logger.info("EC2 error occurred: Message {}, Error Code:{}",
ec2Ex.getMessage(), ec2Ex.awsErrorDetails().errorCode());
            return;
        } else {
            logger.info("An unexpected error occurred: {}",
cause.getMessage());
            return;
        }
    }
    waitForInputToContinue(scanner);
    logger.info(DASHES);

    logger.info(DASHES);
    logger.info("12. Allocate an Elastic IP address and associate it with the
instance.");
    logger.info("""
        An Elastic IP address is a static public IP address that you can
associate with your EC2 instance.
        This allows you to have a fixed, predictable IP address that remains
the same even if your instance
        is stopped, terminated, or replaced.
        This is particularly useful for applications or services that need to
be accessed consistently from a
        known IP address.

        An EC2 Allocation ID (also known as a Reserved Instance Allocation
ID) is a unique identifier associated with a Reserved Instance (RI) that you
have purchased in AWS.

        When you purchase a Reserved Instance, AWS assigns a unique
Allocation ID to it.
        This Allocation ID is used to track and identify the specific RI you
have purchased,
        and it is important for managing and monitoring your Reserved
Instances.
```

```
        """);

        waitForInputToContinue(scanner);
        String allocationId;
        try {
            CompletableFuture<String> future = ec2Actions.allocateAddressAsync();
            allocationId = future.join();
            logger.info("Successfully allocated address with ID: "
+allocationId);
        } catch (RuntimeException rt) {
            Throwable cause = rt.getCause();
            if (cause instanceof Ec2Exception ec2Ex) {
                logger.info("EC2 error occurred: Message {}, Error Code:{}",
ec2Ex.getMessage(), ec2Ex.awsErrorDetails().errorCode());
                return;
            } else {
                logger.info("An unexpected error occurred: {}",
cause.getMessage());
                return;
            }
        }
        logger.info("The allocation Id value is " + allocationId);
        waitForInputToContinue(scanner);
        String associationId;
        try {
            CompletableFuture<String> future =
ec2Actions.associateAddressAsync(newInstanceId, allocationId);
            associationId = future.join();
            logger.info("Successfully associated address with ID: "
+associationId);
        } catch (RuntimeException rt) {
            Throwable cause = rt.getCause();
            if (cause instanceof Ec2Exception ec2Ex) {
                logger.info("EC2 error occurred: Message {}, Error Code:{}",
ec2Ex.getMessage(), ec2Ex.awsErrorDetails().errorCode());
                return;
            } else {
                logger.info("An unexpected error occurred: {}",
cause.getMessage());
                return;
            }
        }
        waitForInputToContinue(scanner);
        logger.info(DASHES);
```

```
        logger.info(DASHES);
        logger.info("13. Describe the instance again. Note that the public IP
address has changed");
        waitForInputToContinue(scanner);
        try {
            CompletableFuture<String> future =
ec2Actions.describeEC2InstancesAsync(newInstanceId);
            publicIp = future.join();
            logger.info("EC2 instance public IP: " + publicIp);
            logger.info("You can SSH to the instance using this command:");
            logger.info("ssh -i " + fileName + " ec2-user@" + publicIp);
        } catch (RuntimeException rt) {
            Throwable cause = rt.getCause();
            if (cause instanceof Ec2Exception ec2Ex) {
                logger.info("EC2 error occurred: Message {}, Error Code:{}",
ec2Ex.getMessage(), ec2Ex.awsErrorDetails().errorCode());
                return;
            } else {
                logger.info("An unexpected error occurred: {}",
cause.getMessage());
                return;
            }
        }
        waitForInputToContinue(scanner);
        logger.info(DASHES);

        logger.info(DASHES);
        logger.info("14. Disassociate and release the Elastic IP address.");
        waitForInputToContinue(scanner);
        try {
            CompletableFuture<DisassociateAddressResponse> future =
ec2Actions.disassociateAddressAsync(associationId);
            future.join();
            logger.info("Address successfully disassociated.");
        } catch (RuntimeException rt) {
            Throwable cause = rt.getCause();
            if (cause instanceof Ec2Exception ec2Ex) {
                // Handle EC2 exceptions.
                logger.info("EC2 error occurred: Message {}, Error Code:{}",
ec2Ex.getMessage(), ec2Ex.awsErrorDetails().errorCode());
                return;
            } else {
```

```
        logger.info("An unexpected error occurred: {}",
cause.getMessage());
        return;
    }
}
waitForInputToContinue(scanner);
try {
    CompletableFuture<ReleaseAddressResponse> future =
ec2Actions.releaseEC2AddressAsync(allocationId);
    future.join(); // Wait for the operation to complete
    logger.info("Elastic IP address successfully released.");
} catch (RuntimeException rte) {
    logger.info("An unexpected error occurred: {}", rte.getMessage());
    return;
}
waitForInputToContinue(scanner);
logger.info(DASHES);

logger.info(DASHES);
logger.info("15. Terminate the instance and use a waiter (this may take a
few mins).");
waitForInputToContinue(scanner);
try {
    CompletableFuture<Object> future =
ec2Actions.terminateEC2Async(newInstanceId);
    future.join();
    logger.info("EC2 instance successfully terminated.");
} catch (RuntimeException rt) {
    Throwable cause = rt.getCause();
    if (cause instanceof Ec2Exception ec2Ex) {
        // Handle EC2 exceptions.
        logger.info("EC2 error occurred: Message {}, Error Code:{}",
ec2Ex.getMessage(), ec2Ex.awsErrorDetails().errorCode());
        return;
    } else {
        logger.info("An unexpected error occurred: {}",
cause.getMessage());
        return;
    }
}
logger.info(DASHES);

logger.info(DASHES);
logger.info("16. Delete the security group.");
```

```
        waitForInputToContinue(scanner);
    try {
        CompletableFuture<Void> future =
ec2Actions.deleteEC2SecGroupAsync(groupId);
        future.join();
        logger.info("Security group successfully deleted.");
    } catch (RuntimeException rt) {
        Throwable cause = rt.getCause();
        if (cause instanceof Ec2Exception ec2Ex) {
            logger.info("EC2 error occurred: Message {}, Error Code:{}",
ec2Ex.getMessage(), ec2Ex.awsErrorDetails().errorCode());
            return;
        } else {
            logger.info("An unexpected error occurred: {}",
cause.getMessage());
            return;
        }
    }
    waitForInputToContinue(scanner);
    logger.info(DASHES);

    logger.info(DASHES);
    logger.info("17. Delete the key.");
    waitForInputToContinue(scanner);
    try {
        CompletableFuture<DeleteKeyPairResponse> future =
ec2Actions.deleteKeysAsync(keyName);
        future.join();
        logger.info("Successfully deleted key pair named " + keyName);
    } catch (RuntimeException rt) {
        Throwable cause = rt.getCause();
        if (cause instanceof Ec2Exception ec2Ex) {
            logger.info("EC2 error occurred: Message {}, Error Code:{}",
ec2Ex.getMessage(), ec2Ex.awsErrorDetails().errorCode());
            return;
        } else {
            logger.info("An unexpected error occurred: {}",
cause.getMessage());
            return;
        }
    }
    waitForInputToContinue(scanner);
    logger.info(DASHES);
```

```
        logger.info(DASHES);
        logger.info("You successfully completed the Amazon EC2 scenario.");
        logger.info(DASHES);
    }
    public static boolean filterName(String name) {
        String[] parts = name.split("/");
        String myValue = parts[4];
        return myValue.contains("amzn2");
    }

    private static void waitForInputToContinue(Scanner scanner) {
        while (true) {
            logger.info("");
            logger.info("Enter 'c' followed by <ENTER> to continue:");
            String input = scanner.nextLine();

            if (input.trim().equalsIgnoreCase("c")) {
                logger.info("Continuing with the program...");
                logger.info("");
                break;
            } else {
                // Handle invalid input.
                logger.info("Invalid input. Please try again.");
            }
        }
    }
}
```

Defina uma classe que envolva EC2 ações.

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.core.client.config.ClientOverrideConfiguration;
import software.amazon.awssdk.http.async.SdkAsyncHttpClient;
import software.amazon.awssdk.http.nio.netty.NettyNioAsyncHttpClient;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.ec2.Ec2AsyncClient;
import software.amazon.awssdk.services.ec2.model.AllocateAddressRequest;
import software.amazon.awssdk.services.ec2.model.AllocateAddressResponse;
import software.amazon.awssdk.services.ec2.model.AssociateAddressRequest;
import software.amazon.awssdk.services.ec2.model.AssociateAddressResponse;
```

```
import
    software.amazon.awssdk.services.ec2.model.AuthorizeSecurityGroupIngressRequest;
import software.amazon.awssdk.services.ec2.model.CreateKeyPairRequest;
import software.amazon.awssdk.services.ec2.model.CreateKeyPairResponse;
import software.amazon.awssdk.services.ec2.model.CreateSecurityGroupRequest;
import software.amazon.awssdk.services.ec2.model.DeleteKeyPairRequest;
import software.amazon.awssdk.services.ec2.model.DeleteKeyPairResponse;
import software.amazon.awssdk.services.ec2.model.DeleteSecurityGroupRequest;
import software.amazon.awssdk.services.ec2.model.DeleteSecurityGroupResponse;
import software.amazon.awssdk.services.ec2.model.DescribeImagesRequest;
import software.amazon.awssdk.services.ec2.model.DescribeInstanceTypesRequest;
import software.amazon.awssdk.services.ec2.model.DescribeInstanceTypesResponse;
import software.amazon.awssdk.services.ec2.model.DescribeInstancesRequest;
import software.amazon.awssdk.services.ec2.model.DescribeKeyPairsResponse;
import software.amazon.awssdk.services.ec2.model.DescribeSecurityGroupsRequest;
import software.amazon.awssdk.services.ec2.model.DescribeSecurityGroupsResponse;
import software.amazon.awssdk.services.ec2.model.DescribeVpcsRequest;
import software.amazon.awssdk.services.ec2.model.DisassociateAddressRequest;
import software.amazon.awssdk.services.ec2.model.DisassociateAddressResponse;
import software.amazon.awssdk.services.ec2.model.DomainType;
import software.amazon.awssdk.services.ec2.model.Ec2Exception;
import software.amazon.awssdk.services.ec2.model.Filter;
import software.amazon.awssdk.services.ec2.model.InstanceTypeInfo;
import software.amazon.awssdk.services.ec2.model.IpPermission;
import software.amazon.awssdk.services.ec2.model.IpRange;
import software.amazon.awssdk.services.ec2.model.ReleaseAddressRequest;
import software.amazon.awssdk.services.ec2.model.ReleaseAddressResponse;
import software.amazon.awssdk.services.ec2.model.RunInstancesRequest;
import software.amazon.awssdk.services.ec2.model.RunInstancesResponse;
import software.amazon.awssdk.services.ec2.model.StopInstancesRequest;
import software.amazon.awssdk.services.ec2.model.StartInstancesRequest;
import software.amazon.awssdk.services.ec2.model.TerminateInstancesRequest;
import software.amazon.awssdk.services.ec2.model.Vpc;
import software.amazon.awssdk.services.ec2.paginators.DescribeImagesPublisher;
import software.amazon.awssdk.services.ec2.paginators.DescribeInstancesPublisher;
import
    software.amazon.awssdk.services.ec2.paginators.DescribeSecurityGroupsPublisher;
import software.amazon.awssdk.services.ec2.paginators.DescribeVpcsPublisher;
import software.amazon.awssdk.services.ec2.waiters.Ec2AsyncWaiter;
import software.amazon.awssdk.services.ssm.SsmAsyncClient;
import software.amazon.awssdk.services.ssm.model.GetParametersByPathRequest;
import software.amazon.awssdk.services.ssm.model.GetParametersByPathResponse;
import software.amazon.awssdk.services.ec2.model.TerminateInstancesResponse;
import java.io.BufferedWriter;
```

```
import java.io.FileWriter;
import java.io.IOException;
import java.time.Duration;
import java.util.List;
import java.util.concurrent.CompletableFuture;
import java.util.concurrent.CompletionException;
import java.util.concurrent.atomic.AtomicReference;

public class EC2Actions {
    private static final Logger logger =
    LoggerFactory.getLogger(EC2Actions.class);
    private static Ec2AsyncClient ec2AsyncClient;

    /**
     * Retrieves an asynchronous Amazon Elastic Container Registry (ECR) client.
     *
     * @return the configured ECR asynchronous client.
     */
    private static Ec2AsyncClient getAsyncClient() {
        if (ec2AsyncClient == null) {
            /**
             * The `NettyNioAsyncHttpClient` class is part of the AWS SDK for Java,
             * version 2,
             * and it is designed to provide a high-performance, asynchronous HTTP
             * client for interacting with AWS services.
             * It uses the Netty framework to handle the underlying network
             * communication and the Java NIO API to
             * provide a non-blocking, event-driven approach to HTTP requests and
             * responses.
             */
            SdkAsyncHttpClient httpClient = NettyNioAsyncHttpClient.builder()
                .maxConcurrency(50) // Adjust as needed.
                .connectionTimeout(Duration.ofSeconds(60)) // Set the connection
                timeout.
                .readTimeout(Duration.ofSeconds(60)) // Set the read timeout.
                .writeTimeout(Duration.ofSeconds(60)) // Set the write timeout.
                .build();

            ClientOverrideConfiguration overrideConfig =
            ClientOverrideConfiguration.builder()
                .apiCallTimeout(Duration.ofMinutes(2)) // Set the overall API
                call timeout.
                .apiCallAttemptTimeout(Duration.ofSeconds(90)) // Set the
                individual call attempt timeout.
        }
    }
}
```



```
        .build();

        ec2AsyncClient = Ec2AsyncClient.builder()
            .region(Region.US_EAST_1)
            .httpClient(httpClient)
            .overrideConfiguration(overrideConfig)
            .build();
    }
    return ec2AsyncClient;
}

/**
 * Deletes a key pair asynchronously.
 *
 * @param keyPair the name of the key pair to delete
 * @return a {@link CompletableFuture} that represents the result of the
asynchronous operation.
 *         The {@link CompletableFuture} will complete with a {@link
DeleteKeyPairResponse} object
 *         that provides the result of the key pair deletion operation.
 */
public CompletableFuture<DeleteKeyPairResponse> deleteKeysAsync(String
keyPair) {
    DeleteKeyPairRequest request = DeleteKeyPairRequest.builder()
        .keyName(keyPair)
        .build();

    // Initiate the asynchronous request to delete the key pair.
    CompletableFuture<DeleteKeyPairResponse> response =
getAsyncClient().deleteKeyPair(request);
    return response.whenComplete((resp, ex) -> {
        if (ex != null) {
            throw new RuntimeException("Failed to delete key pair: " +
keyPair, ex);
        } else if (resp == null) {
            throw new RuntimeException("No response received for deleting key
pair: " + keyPair);
        }
    });
}

/**
 * Deletes an EC2 security group asynchronously.
 *

```

```
    * @param groupId the ID of the security group to delete
    * @return a CompletableFuture that completes when the security group is
    deleted
    */
    public CompletableFuture<Void> deleteEC2SecGroupAsync(String groupId) {
        DeleteSecurityGroupRequest request = DeleteSecurityGroupRequest.builder()
            .groupId(groupId)
            .build();

        CompletableFuture<DeleteSecurityGroupResponse> response =
            getAsyncClient().deleteSecurityGroup(request);
        return response.whenComplete((resp, ex) -> {
            if (ex != null) {
                throw new RuntimeException("Failed to delete security group with
                Id " + groupId, ex);
            } else if (resp == null) {
                throw new RuntimeException("No response received for deleting
                security group with Id " + groupId);
            }
        }).thenApply(resp -> null);
    }

    /**
     * Terminates an EC2 instance asynchronously and waits for it to reach the
     terminated state.
     *
     * @param instanceId the ID of the EC2 instance to terminate
     * @return a {@link CompletableFuture} that completes when the instance has
     been terminated
     * @throws RuntimeException if there is no response from the AWS SDK or if
     there is a failure during the termination process
     */
    public CompletableFuture<Object> terminateEC2Async(String instanceId) {
        TerminateInstancesRequest terminateRequest =
            TerminateInstancesRequest.builder()
                .instanceIds(instanceId)
                .build();

        CompletableFuture<TerminateInstancesResponse> responseFuture =
            getAsyncClient().terminateInstances(terminateRequest);
        return responseFuture.thenCompose(terminateResponse -> {
            if (terminateResponse == null) {
                throw new RuntimeException("No response received for terminating
                instance " + instanceId);
            }
        });
    }
}
```

```
    }
    System.out.println("Going to terminate an EC2 instance and use a
waiter to wait for it to be in terminated state");
    return getAsyncClient().waiter()
        .waitUntilInstanceTerminated(r -> r.instanceIds(instanceId))
        .thenApply(waiterResponse -> null);
}).exceptionally(throwable -> {
    // Handle any exceptions that occurred during the async call
    throw new RuntimeException("Failed to terminate EC2 instance: " +
throwable.getMessage(), throwable);
});
}

/**
 * Releases an Elastic IP address asynchronously.
 *
 * @param allocId the allocation ID of the Elastic IP address to be released
 * @return a {@link CompletableFuture} representing the asynchronous
operation of releasing the Elastic IP address
 */
public CompletableFuture<ReleaseAddressResponse>
releaseEC2AddressAsync(String allocId) {
    ReleaseAddressRequest request = ReleaseAddressRequest.builder()
        .allocationId(allocId)
        .build();

    CompletableFuture<ReleaseAddressResponse> response =
getAsyncClient().releaseAddress(request);
    response.whenComplete((resp, ex) -> {
        if (ex != null) {
            throw new RuntimeException("Failed to release Elastic IP
address", ex);
        }
    });

    return response;
}

/**
 * Disassociates an Elastic IP address from an instance asynchronously.
 *
 * @param associationId The ID of the association you want to disassociate.
 * @return a {@link CompletableFuture} representing the asynchronous
operation of disassociating the address. The
```

```
    *      {@link CompletableFuture} will complete with a {@link
DisassociateAddressResponse} when the operation is
    *      finished.
    *      @throws RuntimeException if the disassociation of the address fails.
    */
    public CompletableFuture<DisassociateAddressResponse>
disassociateAddressAsync(String associationId) {
        Ec2AsyncClient ec2 = getAsyncClient();
        DisassociateAddressRequest addressRequest =
DisassociateAddressRequest.builder()
            .associationId(associationId)
            .build();

        // Disassociate the address asynchronously.
        CompletableFuture<DisassociateAddressResponse> response =
ec2.disassociateAddress(addressRequest);
        response.whenComplete((resp, ex) -> {
            if (ex != null) {
                throw new RuntimeException("Failed to disassociate address", ex);
            }
        });

        return response;
    }

    /**
    * Associates an Elastic IP address with an EC2 instance asynchronously.
    *
    * @param instanceId    the ID of the EC2 instance to associate the Elastic
IP address with
    * @param allocationId  the allocation ID of the Elastic IP address to
associate
    * @return a {@link CompletableFuture} that completes with the association ID
when the operation is successful,
    *         or throws a {@link RuntimeException} if the operation fails
    */
    public CompletableFuture<String> associateAddressAsync(String instanceId,
String allocationId) {
        AssociateAddressRequest associateRequest =
AssociateAddressRequest.builder()
            .instanceId(instanceId)
            .allocationId(allocationId)
            .build();
```

```
        CompletableFuture<AssociateAddressResponse> responseFuture =
getAsyncClient().associateAddress(associateRequest);
        return responseFuture.thenApply(response -> {
            if (response.associationId() != null) {
                return response.associationId();
            } else {
                throw new RuntimeException("Association ID is null after
associating address.");
            }
        }).whenComplete((result, ex) -> {
            if (ex != null) {
                throw new RuntimeException("Failed to associate address", ex);
            }
        });
    }

/**
 * Allocates an Elastic IP address asynchronously in the VPC domain.
 *
 * @return a {@link CompletableFuture} containing the allocation ID of the
allocated Elastic IP address
 */
public CompletableFuture<String> allocateAddressAsync() {
    AllocateAddressRequest allocateRequest = AllocateAddressRequest.builder()
        .domain(DomainType.VPC)
        .build();

    CompletableFuture<AllocateAddressResponse> responseFuture =
getAsyncClient().allocateAddress(allocateRequest);
    return
responseFuture.thenApply(AllocateAddressResponse::allocationId).whenComplete((result,
ex) -> {
        if (ex != null) {
            throw new RuntimeException("Failed to allocate address", ex);
        }
    });
}

/**
 * Asynchronously describes the state of an EC2 instance.
 * The paginator helps you iterate over multiple pages of results.
 *
 * @param newInstanceId the ID of the EC2 instance to describe
```

```
    * @return a {@link CompletableFuture} that, when completed, contains a
    string describing the state of the EC2 instance
    */
    public CompletableFuture<String> describeEC2InstancesAsync(String
newInstanceId) {
        DescribeInstancesRequest request = DescribeInstancesRequest.builder()
            .instanceIds(newInstanceId)
            .build();

        DescribeInstancesPublisher paginator =
getAsyncClient().describeInstancesPaginator(request);
        AtomicReference<String> publicIpAddressRef = new AtomicReference<>();
        return paginator.subscribe(response -> {
            response.reservations().stream()
                .flatMap(reservation -> reservation.instances().stream())
                .filter(instance -> instance.instanceId().equals(newInstanceId))
                .findFirst()
                .ifPresent(instance ->
publicIpAddressRef.set(instance.publicIpAddress()));
        }).thenApply(v -> {
            String publicIpAddress = publicIpAddressRef.get();
            if (publicIpAddress == null) {
                throw new RuntimeException("Instance with ID " + newInstanceId +
" not found.");
            }
            return publicIpAddress;
        }).exceptionally(ex -> {
            logger.info("Failed to describe instances: " + ex.getMessage());
            throw new RuntimeException("Failed to describe instances", ex);
        });
    }

/**
 * Runs an EC2 instance asynchronously.
 *
 * @param instanceType The instance type to use for the EC2 instance.
 * @param keyName The name of the key pair to associate with the EC2
instance.
 * @param groupName The name of the security group to associate with the EC2
instance.
 * @param amiId The ID of the Amazon Machine Image (AMI) to use for the EC2
instance.
 * @return A {@link CompletableFuture} that completes with the ID of the
started EC2 instance.

```

```

    * @throws RuntimeException If there is an error running the EC2 instance.
    */
    public CompletableFuture<String> runInstanceAsync(String instanceType, String
keyName, String groupName, String amiId) {
        RunInstancesRequest runRequest = RunInstancesRequest.builder()
            .instanceType(instanceType)
            .keyName(keyName)
            .securityGroups(groupName)
            .maxCount(1)
            .minCount(1)
            .imageId(amiId)
            .build();

        CompletableFuture<RunInstancesResponse> responseFuture =
getAsyncClient().runInstances(runRequest);
        return responseFuture.thenCompose(response -> {
            String instanceIdVal = response.instances().get(0).instanceId();
            System.out.println("Going to start an EC2 instance and use a waiter
to wait for it to be in running state");
            return getAsyncClient().waiter()
                .waitUntilInstanceExists(r -> r.instanceIds(instanceIdVal))
                .thenCompose(waitResponse -> getAsyncClient().waiter()
                    .waitUntilInstanceRunning(r -> r.instanceIds(instanceIdVal))
                    .thenApply(runningResponse -> instanceIdVal));
        }).exceptionally(throwable -> {
            // Handle any exceptions that occurred during the async call
            throw new RuntimeException("Failed to run EC2 instance: " +
throwable.getMessage(), throwable);
        });
    }

    /**
     * Asynchronously retrieves the instance types available in the current AWS
region.
     * <p>
     * This method uses the AWS SDK's asynchronous API to fetch the available
instance types
     * and then processes the response. It logs the memory information, network
information,
     * and instance type for each instance type returned. Additionally, it
returns a
     * {@link CompletableFuture} that resolves to the instance type string for
the "t2.2xlarge"

```

```
    * instance type, if it is found in the response. If the "t2.2xlarge"
instance type is not
    * found, an empty string is returned.
    * </p>
    *
    * @return a {@link CompletableFuture} that resolves to the instance type
string for the
    * "t2.2xlarge" instance type, or an empty string if the instance type is not
found
    */
    public CompletableFuture<String> getInstanceTypesAsync() {
        DescribeInstanceTypesRequest typesRequest =
DescribeInstanceTypesRequest.builder()
            .maxResults(10)
            .build();

        CompletableFuture<DescribeInstanceTypesResponse> response =
getAsyncClient().describeInstanceTypes(typesRequest);
        response.whenComplete((resp, ex) -> {
            if (resp != null) {
                List<InstanceTypeInfo> instanceTypes = resp.instanceTypes();
                for (InstanceTypeInfo type : instanceTypes) {
                    logger.info("The memory information of this type is " +
type.memoryInfo().sizeInMiB());
                    logger.info("Network information is " +
type.networkInfo().toString());
                    logger.info("Instance type is " +
type.instanceType().toString());
                }
            } else {
                throw (RuntimeException) ex;
            }
        });

        return response.thenApply(resp -> {
            for (InstanceTypeInfo type : resp.instanceTypes()) {
                String instanceType = type.instanceType().toString();
                if (instanceType.equals("t2.2xlarge")) {
                    return instanceType;
                }
            }
            return "";
        });
    }
}
```



```
/**
 * Asynchronously describes an AWS EC2 image with the specified image ID.
 *
 * @param imageId the ID of the image to be described
 * @return a {@link CompletableFuture} that, when completed, contains the ID
of the described image
 * @throws RuntimeException if no images are found with the provided image
ID, or if an error occurs during the AWS API call
 */
public CompletableFuture<String> describeImageAsync(String imageId) {
    DescribeImagesRequest imagesRequest = DescribeImagesRequest.builder()
        .imageIds(imageId)
        .build();

    AtomicReference<String> imageIdRef = new AtomicReference<>();
    DescribeImagesPublisher paginator =
getAsyncClient().describeImagesPaginator(imagesRequest);
    return paginator.subscribe(response -> {
        response.images().stream()
            .filter(image -> image.imageId().equals(imageId))
            .findFirst()
            .ifPresent(image -> {
                logger.info("The description of the image is " +
image.description());
                logger.info("The name of the image is " + image.name());
                imageIdRef.set(image.imageId());
            });
    }).thenApply(v -> {
        String id = imageIdRef.get();
        if (id == null) {
            throw new RuntimeException("No images found with the provided
image ID.");
        }
        return id;
    }).exceptionally(ex -> {
        logger.info("Failed to describe image: " + ex.getMessage());
        throw new RuntimeException("Failed to describe image", ex);
    });
}

/**
 * Retrieves the parameter values asynchronously using the AWS Systems
Manager (SSM) API.
```

```
*
 * @return a {@link CompletableFuture} that holds the response from the SSM
API call to get parameters by path
 */
public CompletableFuture<GetParametersByPathResponse> getParaValuesAsync() {
    SsmAsyncClient ssmClient = SsmAsyncClient.builder()
        .region(Region.US_EAST_1)
        .build();

    GetParametersByPathRequest parameterRequest =
GetParametersByPathRequest.builder()
    .path("/aws/service/ami-amazon-linux-latest")
    .build();

    // Create a CompletableFuture to hold the final result.
    CompletableFuture<GetParametersByPathResponse> responseFuture = new
CompletableFuture<>();
    ssmClient.getParametersByPath(parameterRequest)
        .whenComplete((response, exception) -> {
            if (exception != null) {
                responseFuture.completeExceptionally(new
RuntimeException("Failed to get parameters by path", exception));
            } else {
                responseFuture.complete(response);
            }
        });

    return responseFuture;
}

/**
 * Asynchronously describes the security groups for the specified group ID.
 *
 * @param groupName the name of the security group to describe
 * @return a {@link CompletableFuture} that represents the asynchronous
operation
 *
 * of describing the security groups. The future will complete with a
 *
 * {@link DescribeSecurityGroupsResponse} object that contains the
 *
 * security group information.
 */
public CompletableFuture<String> describeSecurityGroupArnByNameAsync(String
groupName) {
```

```

    DescribeSecurityGroupsRequest request =
DescribeSecurityGroupsRequest.builder()
    .groupNames(groupName)
    .build();

    DescribeSecurityGroupsPublisher paginator =
getClient().describeSecurityGroupsPaginator(request);
    AtomicReference<String> groupIdRef = new AtomicReference<>();
    return paginator.subscribe(response -> {
        response.securityGroups().stream()
            .filter(securityGroup ->
securityGroup.groupName().equals(groupName))
            .findFirst()
            .ifPresent(securityGroup ->
groupIdRef.set(securityGroup.groupId()));
    }).thenApply(v -> {
        String groupId = groupIdRef.get();
        if (groupId == null) {
            throw new RuntimeException("No security group found with the
name: " + groupName);
        }
        return groupId;
    }).exceptionally(ex -> {
        logger.info("Failed to describe security group: " + ex.getMessage());
        throw new RuntimeException("Failed to describe security group", ex);
    });
}

/**
 * Creates a new security group asynchronously with the specified group name,
description, and VPC ID. It also
 * authorizes inbound traffic on ports 80 and 22 from the specified IP
address.
 *
 * @param groupName    the name of the security group to create
 * @param groupDesc    the description of the security group
 * @param vpcId        the ID of the VPC in which to create the security
group
 * @param myIpAddress  the IP address from which to allow inbound traffic
(e.g., "192.168.1.1/0" to allow traffic from
 *                    any IP address in the 192.168.1.0/24 subnet)
 * @return a CompletableFuture that, when completed, returns the ID of the
created security group

```

```
    * @throws RuntimeException if there was a failure creating the security
    group or authorizing the inbound traffic
    */
    public CompletableFuture<String> createSecurityGroupAsync(String groupName,
String groupDesc, String vpcId, String myIpAddress) {
        CreateSecurityGroupRequest createRequest =
CreateSecurityGroupRequest.builder()
            .groupName(groupName)
            .description(groupDesc)
            .vpcId(vpcId)
            .build();

        return getAsyncClient().createSecurityGroup(createRequest)
            .thenCompose(createResponse -> {
                String groupId = createResponse.groupId();
                IpRange ipRange = IpRange.builder()
                    .cidrIp(myIpAddress + "/32")
                    .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();

                return
getAsyncClient().authorizeSecurityGroupIngress(authRequest)
                    .thenApply(authResponse -> groupId);
            })
            .whenComplete((result, exception) -> {
```

```
        if (exception != null) {
            if (exception instanceof CompletionException &&
exception.getCause() instanceof Ec2Exception) {
                throw (Ec2Exception) exception.getCause();
            } else {
                throw new RuntimeException("Failed to create security
group: " + exception.getMessage(), exception);
            }
        }
    });
}

/**
 * Asynchronously describes the key pairs associated with the current AWS
account.
 *
 * @return a {@link CompletableFuture} containing the {@link
DescribeKeyPairsResponse} object, which provides
 * information about the key pairs.
 */
public CompletableFuture<DescribeKeyPairsResponse> describeKeysAsync() {
    CompletableFuture<DescribeKeyPairsResponse> responseFuture =
getAsyncClient().describeKeyPairs();
    responseFuture.whenComplete((response, exception) -> {
        if (exception != null) {
            throw new RuntimeException("Failed to describe key pairs: " +
exception.getMessage(), exception);
        }
    });

    return responseFuture;
}

/**
 * Creates a new key pair asynchronously.
 *
 * @param keyName the name of the key pair to create
 * @param fileName the name of the file to write the key material to
 * @return a {@link CompletableFuture} that represents the asynchronous
operation
 * of creating the key pair and writing the key material to a file
 */
public CompletableFuture<CreateKeyPairResponse> createKeyPairAsync(String
keyName, String fileName) {
```

```
        CreateKeyPairRequest request = CreateKeyPairRequest.builder()
            .keyName(keyName)
            .build();

        CompletableFuture<CreateKeyPairResponse> responseFuture =
getAsyncClient().createKeyPair(request);
        responseFuture.whenComplete((response, exception) -> {
            if (response != null) {
                try {
                    BufferedWriter writer = new BufferedWriter(new
FileWriter(fileName));
                    writer.write(response.keyMaterial());
                    writer.close();
                } catch (IOException e) {
                    throw new RuntimeException("Failed to write key material to
file: " + e.getMessage(), e);
                }
            } else {
                throw new RuntimeException("Failed to create key pair: " +
exception.getMessage(), exception);
            }
        });

        return responseFuture;
    }

    /**
     * Describes the first default VPC asynchronously and using a paginator.
     *
     * @return a {@link CompletableFuture} that, when completed, contains the
first default VPC found.
     */
    public CompletableFuture<Vpc> describeFirstEC2VpcAsync() {
        Filter myFilter = Filter.builder()
            .name("is-default")
            .values("true")
            .build();

        DescribeVpcsRequest request = DescribeVpcsRequest.builder()
            .filters(myFilter)
            .build();

        DescribeVpcsPublisher paginator =
getAsyncClient().describeVpcsPaginator(request);
```

```

    AtomicReference<Vpc> vpcRef = new AtomicReference<>();
    return paginator.subscribe(response -> {
        response.vpcs().stream()
            .findFirst()
            .ifPresent(vpcRef::set);
    }).thenApply(v -> {
        Vpc vpc = vpcRef.get();
        if (vpc == null) {
            throw new RuntimeException("Default VPC not found");
        }
        return vpc;
    }).exceptionally(ex -> {
        logger.info("Failed to describe VPCs: " + ex.getMessage());
        throw new RuntimeException("Failed to describe VPCs", ex);
    });
}

/**
 * Stops the EC2 instance with the specified ID asynchronously and waits for
the instance to stop.
 *
 * @param instanceId the ID of the EC2 instance to stop
 * @return a {@link CompletableFuture} that completes when the instance has
been stopped, or exceptionally if an error occurs
 */
public CompletableFuture<Void> stopInstanceAsync(String instanceId) {
    StopInstancesRequest stopRequest = StopInstancesRequest.builder()
        .instanceIds(instanceId)
        .build();

    DescribeInstancesRequest describeRequest =
DescribeInstancesRequest.builder()
        .instanceIds(instanceId)
        .build();

    Ec2AsyncWaiter ec2Waiter = Ec2AsyncWaiter.builder()
        .client(getAsyncClient())
        .build();

    CompletableFuture<Void> resultFuture = new CompletableFuture<>();
    logger.info("Stopping instance " + instanceId + " and waiting for it to
stop.");
    getAsyncClient().stopInstances(stopRequest)
        .thenCompose(response -> {

```

```
        if (response.stoppingInstances().isEmpty()) {
            return CompletableFuture.failedFuture(new
RuntimeException("No instances were stopped. Please check the instance ID: " +
instanceId));
        }
        return ec2Waiter.waitUntilInstanceStopped(describeRequest);
    })
    .thenAccept(waiterResponse -> {
        logger.info("Successfully stopped instance " + instanceId);
        resultFuture.complete(null);
    })
    .exceptionally(throwable -> {
        logger.error("Failed to stop instance " + instanceId + ": " +
throwable.getMessage(), throwable);
        resultFuture.completeExceptionally(new RuntimeException("Failed
to stop instance: " + throwable.getMessage(), throwable));
        return null;
    });
    });

    return resultFuture;
}

/**
 * Starts an Amazon EC2 instance asynchronously and waits until it is in the
"running" state.
 *
 * @param instanceId the ID of the instance to start
 * @return a {@link CompletableFuture} that completes when the instance has
been started and is in the "running" state, or exceptionally if an error occurs
 */
public CompletableFuture<Void> startInstanceAsync(String instanceId) {
    StartInstancesRequest startRequest = StartInstancesRequest.builder()
        .instanceIds(instanceId)
        .build();

    Ec2AsyncWaiter ec2Waiter = Ec2AsyncWaiter.builder()
        .client(getAsyncClient())
        .build();

    DescribeInstancesRequest describeRequest =
DescribeInstancesRequest.builder()
        .instanceIds(instanceId)
        .build();
```



```
        logger.info("Starting instance " + instanceId + " and waiting for it to
run.");
        CompletableFuture<Void> resultFuture = new CompletableFuture<>();
        return getAsyncClient().startInstances(startRequest)
            .thenCompose(response ->
                ec2Waiter.waitForInstanceRunning(describeRequest)
            )
            .thenAccept(waiterResponse -> {
                logger.info("Successfully started instance " + instanceId);
                resultFuture.complete(null);
            })
            .exceptionally(throwable -> {
                resultFuture.completeExceptionally(new RuntimeException("Failed
to start instance: " + throwable.getMessage(), throwable));
                return null;
            });
    }
}
```

- Para API obter detalhes, consulte os tópicos a seguir em AWS SDK for Java 2.x APIReferência.
 - [AllocateAddress](#)
 - [AssociateAddress](#)
 - [AuthorizeSecurityGroupIngress](#)
 - [CreateKeyPair](#)
 - [CreateSecurityGroup](#)
 - [DeleteKeyPair](#)
 - [DeleteSecurityGroup](#)
 - [DescribeImages](#)
 - [DescribeInstanceTypes](#)
 - [DescribeInstances](#)
 - [DescribeKeyPairs](#)
 - [DescribeSecurityGroups](#)
 - [DisassociateAddress](#)
 - [ReleaseAddress](#)

- [RunInstances](#)
- [StartInstances](#)
- [StopInstances](#)
- [TerminateInstances](#)
- [UnmonitorInstances](#)

JavaScript

SDK para JavaScript (v3)

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

Esse arquivo contém uma lista de ações comuns usadas com EC2. As etapas são construídas com uma estrutura de cenário que simplifica a execução de um exemplo interativo. Para ver o contexto completo, visite o GitHub repositório.

```
import { tmpdir } from "node:os";
import { writeFile, mkdtemp, rm } from "node:fs/promises";
import { join } from "node:path";
import { get } from "node:http";

import {
  AllocateAddressCommand,
  AssociateAddressCommand,
  AuthorizeSecurityGroupIngressCommand,
  CreateKeyPairCommand,
  CreateSecurityGroupCommand,
  DeleteKeyPairCommand,
  DeleteSecurityGroupCommand,
  DisassociateAddressCommand,
  paginateDescribeImages,
  paginateDescribeInstances,
  paginateDescribeInstanceTypes,
  ReleaseAddressCommand,
  RunInstancesCommand,
```

```
StartInstancesCommand,
StopInstancesCommand,
TerminateInstancesCommand,
waitUntilInstanceStatusOk,
waitUntilInstanceStopped,
waitUntilInstanceTerminated,
} from "@aws-sdk/client-ec2";

import {
  ScenarioAction,
  ScenarioInput,
  ScenarioOutput,
} from "@aws-doc-sdk-examples/lib/scenario/index.js";

import { paginateGetParametersByPath, SSMClient } from "@aws-sdk/client-ssm";

/**
 * @typedef {{
 *   ec2Client: import('@aws-sdk/client-ec2').EC2Client,
 *   errors: Error[],
 *   keyPairId?: string,
 *   tmpDirectory?: string,
 *   securityGroupId?: string,
 *   ipAddress?: string,
 *   images?: import('@aws-sdk/client-ec2').Image[],
 *   image?: import('@aws-sdk/client-ec2').Image,
 *   instanceTypes?: import('@aws-sdk/client-ec2').InstanceTypeInfo[],
 *   instanceId?: string,
 *   instanceIpAddress?: string,
 *   allocationId?: string,
 *   allocatedIpAddress?: string,
 *   associationId?: string,
 * }} State
 */

/**
 * A skip function provided to the `skipWhen` of a Step when you want
 * to ignore that step if any errors have occurred.
 * @param {State} state
 */
const skipWhenErrors = (state) => state.errors.length > 0;

const MAX_WAITER_TIME_IN_SECONDS = 60 * 8;
```

```
export const confirm = new ScenarioInput("confirmContinue", "Continue?", {
  type: "confirm",
  skipWhen: skipWhenErrors,
});
```

```
export const exitOnNoConfirm = new ScenarioAction(
  "exitOnConfirmContinueFalse",
  (** @type { { earlyExit: boolean } & Record<string, any>} */ state) => {
    if (!state[confirm.name]) {
      state.earlyExit = true;
    }
  },
  {
    skipWhen: skipWhenErrors,
  },
);
```

```
export const greeting = new ScenarioOutput(
  "greeting",
  `
```

Welcome to the Amazon EC2 basic usage scenario.

Before you launch an instances, you'll need to provide a few things:

- A key pair - This is for SSH access to your EC2 instance. You only need to provide the name.
- A security group - This is used for configuring access to your instance. Again, only the name is needed.
- An IP address - Your public IP address will be fetched.
- An Amazon Machine Image (AMI)
- A compatible instance type`,

```
{ header: true, preformatted: true, skipWhen: skipWhenErrors },
);
```

```
export const provideKeyName = new ScenarioInput(
  "keyPairName",
  "Provide a name for a new key pair.",
  { type: "input", default: "ec2-example-key-pair", skipWhen: skipWhenErrors },
);
```

```
export const createKeyPair = new ScenarioAction(
  "createKeyPair",
  async (** @type {State} */ state) => {
    try {
```

```
// Create a key pair in Amazon EC2.
const { KeyMaterial, KeyPairId } = await state.ec2Client.send(
  // A unique name for the key pair. Up to 255 ASCII characters.
  new CreateKeyPairCommand({ KeyName: state[provideKeyPairName.name] }),
);

state.keyPairId = KeyPairId;

// Save the private key in a temporary location.
state.tmpDirectory = await mkdtemp(join(tmpdir(), "ec2-scenario-tmp"));
await writeFile(
  `${state.tmpDirectory}/${state[provideKeyPairName.name]}.pem`,
  KeyMaterial,
  {
    mode: 0o400,
  },
);
} catch (caught) {
  if (
    caught instanceof Error &&
    caught.name === "InvalidKeyPair.Duplicate"
  ) {
    caught.message = `${caught.message}. Try another key name.`;
  }

  state.errors.push(caught);
}
},
{ skipWhen: skipWhenErrors },
);

export const logKeyPair = new ScenarioOutput(
  "logKeyPair",
  (/** @type {State} */ state) =>
    `Created the key pair ${state[provideKeyPairName.name]}.`,
  { skipWhen: skipWhenErrors },
);

export const confirmDeleteKeyPair = new ScenarioInput(
  "confirmDeleteKeyPair",
  "Do you want to delete the key pair?",
  {
    type: "confirm",
    // Don't do anything when a key pair was never created.
  }
);
```

```

    skipWhen: (/** @type {State} */ state) => !state.keyPairId,
  },
);

export const maybeDeleteKeyPair = new ScenarioAction(
  "deleteKeyPair",
  async (/** @type {State} */ state) => {
    try {
      // Delete a key pair by name from EC2
      await state.ec2Client.send(
        new DeleteKeyPairCommand({ KeyName: state[provideKeyPairName.name] }),
      );
    } catch (caught) {
      if (
        caught instanceof Error &&
        // Occurs when a required parameter (e.g. KeyName) is undefined.
        caught.name === "MissingParameter"
      ) {
        caught.message = `${caught.message}. Did you provide the required value?`;
      }
      state.errors.push(caught);
    }
  },
  {
    // Don't do anything when there's no key pair to delete or the user chooses
    // to keep it.
    skipWhen: (/** @type {State} */ state) =>
      !state.keyPairId || !state[confirmDeleteKeyPair.name],
  },
);

export const provideSecurityGroupName = new ScenarioInput(
  "securityGroupName",
  "Provide a name for a new security group.",
  { type: "input", default: "ec2-scenario-sg", skipWhen: skipWhenErrors },
);

export const createSecurityGroup = new ScenarioAction(
  "createSecurityGroup",
  async (/** @type {State} */ state) => {
    try {
      // Create a new security group that will be used to configure ingress/
      egress for

```

```
// an EC2 instance.
const { GroupId } = await state.ec2Client.send(
  new CreateSecurityGroupCommand({
    GroupName: state[provideSecurityGroupName.name],
    Description: "A security group for the Amazon EC2 example.",
  }),
);
state.securityGroupId = GroupId;
} catch (caught) {
  if (caught instanceof Error && caught.name === "InvalidGroup.Duplicate") {
    caught.message = `${caught.message}. Please provide a different name for
your security group.`;
  }

  state.errors.push(caught);
},
{ skipWhen: skipWhenErrors },
);

export const logSecurityGroup = new ScenarioOutput(
  "logSecurityGroup",
  (/** @type {State} */ state) =>
    `Created the security group ${state.securityGroupId}.`,
  { skipWhen: skipWhenErrors },
);

export const confirmDeleteSecurityGroup = new ScenarioInput(
  "confirmDeleteSecurityGroup",
  "Do you want to delete the security group?",
  {
    type: "confirm",
    // Don't do anything when a security group was never created.
    skipWhen: (/** @type {State} */ state) => !state.securityGroupId,
  },
);

export const maybeDeleteSecurityGroup = new ScenarioAction(
  "deleteSecurityGroup",
  async (/** @type {State} */ state) => {
    try {
      // Delete the security group if the 'skipWhen' condition below is not met.
      await state.ec2Client.send(
        new DeleteSecurityGroupCommand({
```

```

        GroupId: state.securityGroupId,
      )),
    );
  } catch (caught) {
    if (
      caught instanceof Error &&
      caught.name === "InvalidGroupId.Malformed"
    ) {
      caught.message = `${caught.message}. Please provide a valid GroupId.`;
    }
    state.errors.push(caught);
  }
},
{
  // Don't do anything when there's no security group to delete
  // or the user chooses to keep it.
  skipWhen: (/** @type {State} */ state) =>
    !state.securityGroupId || !state[confirmDeleteSecurityGroup.name],
},
);

export const authorizeSecurityGroupIngress = new ScenarioAction(
  "authorizeSecurity",
  async (/** @type {State} */ state) => {
    try {
      // Get the public IP address of the machine running this example.
      const ipAddress = await 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);
        });
      });
      state.ipAddress = ipAddress;
      // Allow ingress from the IP address above to the security group.
      // This will allow you to SSH into the EC2 instance.
      const command = new AuthorizeSecurityGroupIngressCommand({
        GroupId: state.securityGroupId,
        IpPermissions: [
          {

```



```

        IpProtocol: "tcp",
        FromPort: 22,
        ToPort: 22,
        IpRanges: [{ CidrIp: `${ipAddress}/32` }],
    },
],
});

    await state.ec2Client.send(command);
} catch (caught) {
    if (
        caught instanceof Error &&
        caught.name === "InvalidGroupId.Malformed"
    ) {
        caught.message = `${caught.message}. Please provide a valid GroupId.`;
    }

    state.errors.push(caught);
}
},
{ skipWhen: skipWhenErrors },
);

export const logSecurityGroupIngress = new ScenarioOutput(
    "logSecurityGroupIngress",
    (/** @type {State} */ state) =>
        `Allowed SSH access from your public IP: ${state.ipAddress}.`,
    { skipWhen: skipWhenErrors },
);

export const getImages = new ScenarioAction(
    "images",
    async (/** @type {State} */ state) => {
        const AMIs = [];
        // Some AWS services publish information about common artifacts as AWS
        Systems Manager (SSM)
        // public parameters. For example, the Amazon Elastic Compute Cloud (Amazon
        EC2)
        // service publishes information about Amazon Machine Images (AMIs) as public
        parameters.

        // Create the paginator for getting images. Actions that return multiple
        pages of
        // results have paginators to simplify those calls.

```

```
const getParametersByPathPaginator = paginateGetParametersByPath(
  {
    // Not storing this client in state since it's only used once.
    client: new SSMClient({}),
  },
  {
    // The path to the public list of the latest amazon-linux instances.
    Path: "/aws/service/ami-amazon-linux-latest",
  },
);

try {
  for await (const page of getParametersByPathPaginator) {
    for (const param of page.Parameters) {
      // Filter by Amazon Linux 2
      if (param.Name.includes("amzn2")) {
        AMIs.push(param.Value);
      }
    }
  }
} catch (caught) {
  if (caught instanceof Error && caught.name === "InvalidFilterValue") {
    caught.message = `${caught.message} Please provide a valid filter value
for paginateGetParametersByPath.`;
  }
  state.errors.push(caught);
  return;
}

const imageDetails = [];
const describeImagesPaginator = paginateDescribeImages(
  { client: state.ec2Client },
  // The images found from the call to SSM.
  { ImageIds: AMIs },
);

try {
  // Get more details for the images found above.
  for await (const page of describeImagesPaginator) {
    imageDetails.push...(page.Images || []);
  }

  // Store the image details for later use.
  state.images = imageDetails;
}
```

```
    } catch (caught) {
      if (caught instanceof Error && caught.name === "InvalidAMIID.NotFound") {
        caught.message = `${caught.message}. Please provide a valid image id.`;
      }

      state.errors.push(caught);
    }
  },
  { skipWhen: skipWhenErrors },
);

export const provideImage = new ScenarioInput(
  "image",
  "Select one of the following images.",
  {
    type: "select",
    choices: (/** @type { State } */ state) =>
      state.images.map((image) => ({
        name: `${image.Description}`,
        value: image,
      })),
    default: (/** @type { State } */ state) => state.images[0],
    skipWhen: skipWhenErrors,
  },
);

export const getCompatibleInstanceTypes = new ScenarioAction(
  "getCompatibleInstanceTypes",
  async (/** @type {State} */ state) => {
    // Get more details about instance types that match the architecture of
    // the provided image.
    const paginator = paginateDescribeInstanceTypes(
      { client: state.ec2Client, pageSize: 25 },
      {
        Filters: [
          {
            Name: "processor-info.supported-architecture",
            // The value selected from provideImage()
            Values: [state.image.Architecture],
          },
          // Filter for smaller, less expensive, types.
          { Name: "instance-type", Values: ["*.micro", "*.small"] },
        ],
      },
    );
  },
);
```

```
);

const instanceTypes = [];

try {
  for await (const page of paginator) {
    if (page.InstanceTypes.length) {
      instanceTypes.push(...(page.InstanceTypes || []));
    }
  }

  if (!instanceTypes.length) {
    state.errors.push(
      "No instance types matched the instance type filters.",
    );
  }
} catch (caught) {
  if (caught instanceof Error && caught.name === "InvalidParameterValue") {
    caught.message = `${caught.message}. Please check the provided values and
try again.`;
  }

  state.errors.push(caught);
}

state.instanceTypes = instanceTypes;
},
{ skipWhen: skipWhenErrors },
);

export const provideInstanceType = new ScenarioInput(
  "instanceType",
  "Select an instance type.",
  {
    choices: (/** @type {State} */ state) =>
      state.instanceTypes.map((instanceType) => ({
        name: `${instanceType.InstanceType} - Memory:
${instanceType.MemoryInfo.SizeInMiB}`,
        value: instanceType.InstanceType,
      })),
    type: "select",
    default: (/** @type {State} */ state) =>
      state.instanceTypes[0].InstanceType,
    skipWhen: skipWhenErrors,
  }
);
```

```
    },
  );

export const runInstance = new ScenarioAction(
  "runInstance",
  async (** @type { State } */ state) => {
    const { Instances } = await state.ec2Client.send(
      new RunInstancesCommand({
        KeyName: state[provideKeyPairName.name],
        SecurityGroupIds: [state.securityGroupId],
        ImageId: state.image.ImageId,
        InstanceType: state[provideInstanceType.name],
        // Availability Zones have capacity limitations that may impact your
        // ability to launch instances.
        // The `RunInstances` operation will only succeed if it can allocate at
        // least the `MinCount` of instances.
        // However, EC2 will attempt to launch up to the `MaxCount` of instances,
        // even if the full request cannot be satisfied.
        // If you need a specific number of instances, use `MinCount` and
        // `MaxCount` set to the same value.
        // If you want to launch up to a certain number of instances, use
        // `MaxCount` and let EC2 provision as many as possible.
        // If you require a minimum number of instances, but do not want to
        // exceed a maximum, use both `MinCount` and `MaxCount`.
        MinCount: 1,
        MaxCount: 1,
      })),
  );

state.instanceId = Instances[0].InstanceId;

try {
  // Poll `DescribeInstanceStatus` until status is "ok".
  await waitUntilInstanceStatusOk(
    {
      client: state.ec2Client,
      maxWaitTime: MAX_WAITER_TIME_IN_SECONDS,
    },
    { InstanceIds: [Instances[0].InstanceId] },
  );
} catch (caught) {
  if (caught instanceof Error && caught.name === "TimeoutError") {
    caught.message = `${caught.message}. Try increasing the maxWaitTime in
the waiter.`;
  }
}
```

```
    }

    state.errors.push(caught);
  }
},
{ skipWhen: skipWhenErrors },
);

export const logRunInstance = new ScenarioOutput(
  "logRunInstance",
  "The next step is to run your EC2 instance for the first time. This can take a
  few minutes.",
  { header: true, skipWhen: skipWhenErrors },
);

export const describeInstance = new ScenarioAction(
  "describeInstance",
  async (** @type { State } */ state) => {
    /** @type { import("@aws-sdk/client-ec2").Instance[] } */
    const instances = [];

    try {
      const paginator = paginateDescribeInstances(
        {
          client: state.ec2Client,
        },
        {
          // Only get our created instance.
          InstanceIds: [state.instanceId],
        },
      );
    }

    for await (const page of paginator) {
      for (const reservation of page.Reservations) {
        instances.push(...reservation.Instances);
      }
    }
    if (instances.length !== 1) {
      throw new Error(`Instance ${state.instanceId} not found.`);
    }

    // The only info we need is the IP address for SSH purposes.
    state.instanceIpAddress = instances[0].PublicIpAddress;
  } catch (caught) {
```

```
    if (caught instanceof Error && caught.name === "InvalidParameterValue") {
      caught.message = `${caught.message}. Please check provided values and try
again.`;
    }

    state.errors.push(caught);
  }
},
{ skipWhen: skipWhenErrors },
);

export const logSSHConnectionInfo = new ScenarioOutput(
  "logSSHConnectionInfo",
  (/** @type { State } */ state) =>
    `You can now SSH into your instance using the following command:
ssh -i ${state.tmpDirectory}/${state[provideKeyPairName.name]}.pem ec2-user@
${state.instanceIpAddress}`,
  { preformatted: true, skipWhen: skipWhenErrors },
);

export const logStopInstance = new ScenarioOutput(
  "logStopInstance",
  "Stopping your EC2 instance.",
  { skipWhen: skipWhenErrors },
);

export const stopInstance = new ScenarioAction(
  "stopInstance",
  async (/** @type { State } */ state) => {
    try {
      await state.ec2Client.send(
        new StopInstancesCommand({
          InstanceIds: [state.instanceId],
        }),
      );
    };

    await waitUntilInstanceStopped(
      {
        client: state.ec2Client,
        maxWaitTime: MAX_WAITER_TIME_IN_SECONDS,
      },
      { InstanceIds: [state.instanceId] },
    );
  } catch (caught) {
```

```
    if (caught instanceof Error && caught.name === "TimeoutError") {
      caught.message = `${caught.message}. Try increasing the maxWaitTime in
the waiter.`;
    }

    state.errors.push(caught);
  }
},
// Don't try to stop an instance that doesn't exist.
{ skipWhen: (/** @type { State } */ state) => !state.instanceId },
);

export const logIpAddressBehavior = new ScenarioOutput(
  "logIpAddressBehavior",
  [
    "When you run an instance, by default it's assigned an IP address.",
    "That IP address is not static. It will change every time the instance is
restarted.",
    "The next step is to stop and restart your instance to demonstrate this
behavior.",
  ].join(" "),
  { header: true, skipWhen: skipWhenErrors },
);

export const logStartInstance = new ScenarioOutput(
  "logStartInstance",
  (/** @type { State } */ state) => `Starting instance ${state.instanceId}`,
  { skipWhen: skipWhenErrors },
);

export const startInstance = new ScenarioAction(
  "startInstance",
  async (/** @type { State } */ state) => {
    try {
      await state.ec2Client.send(
        new StartInstancesCommand({
          InstanceIds: [state.instanceId],
        }),
      );
    }

    await waitUntilInstanceStatusOk(
      {
        client: state.ec2Client,
        maxWaitTime: MAX_WAITER_TIME_IN_SECONDS,
      }
    );
  }
);
```



```
    },
    { InstanceIds: [state.instanceId] },
  );
} catch (caught) {
  if (caught instanceof Error && caught.name === "TimeoutError") {
    caught.message = `${caught.message}. Try increasing the maxWaitTime in
the waiter.`;
  }

  state.errors.push(caught);
}
},
{ skipWhen: skipWhenErrors },
);

export const logIpAllocation = new ScenarioOutput(
  "logIpAllocation",
  [
    "It is possible to have a static IP address.",
    "To demonstrate this, an IP will be allocated and associated to your EC2
instance.",
  ].join(" "),
  { header: true, skipWhen: skipWhenErrors },
);

export const allocateIp = new ScenarioAction(
  "allocateIp",
  async (** @type { State } */ state) => {
    try {
      // An Elastic IP address is allocated to your AWS account, and is yours
until you release it.
      const { AllocationId, PublicIp } = await state.ec2Client.send(
        new AllocateAddressCommand({}),
      );
      state.allocationId = AllocationId;
      state.allocatedIpAddress = PublicIp;
    } catch (caught) {
      if (caught instanceof Error && caught.name === "MissingParameter") {
        caught.message = `${caught.message}. Did you provide these values?`;
      }
      state.errors.push(caught);
    }
  },
  { skipWhen: skipWhenErrors },
);
```

```
);

export const associateIp = new ScenarioAction(
  "associateIp",
  async (** @type { State } */ state) => {
    try {
      // Associate an allocated IP address to an EC2 instance. An IP address can
      // be allocated
      // with the AllocateAddress action.
      const { AssociationId } = await state.ec2Client.send(
        new AssociateAddressCommand({
          AllocationId: state.allocationId,
          InstanceId: state.instanceId,
        })),
    );
    state.associationId = AssociationId;
    // Update the IP address that is being tracked to match
    // the one just associated.
    state.instanceIpAddress = state.allocatedIpAddress;
  } catch (caught) {
    if (
      caught instanceof Error &&
      caught.name === "InvalidAllocationID.NotFound"
    ) {
      caught.message = `${caught.message}. Did you provide the ID of a valid
      Elastic IP address AllocationId?`;
    }
    state.errors.push(caught);
  }
},
{ skipWhen: skipWhenErrors },
);

export const logStaticIpProof = new ScenarioOutput(
  "logStaticIpProof",
  "The IP address should remain the same even after stopping and starting the
  instance.",
  { header: true, skipWhen: skipWhenErrors },
);

export const logCleanUp = new ScenarioOutput(
  "logCleanUp",
  "That's it! You can choose to clean up the resources now, or clean them up on
  your own later.",
```

```
    { header: true, skipWhen: skipWhenErrors },
  );

export const confirmDisassociateAddress = new ScenarioInput(
  "confirmDisassociateAddress",
  "Do you want to disassociate and release the static IP address created
  earlier?",
  {
    type: "confirm",
    skipWhen: (/** @type { State } */ state) => !state.associationId,
  },
);

export const maybeDisassociateAddress = new ScenarioAction(
  "maybeDisassociateAddress",
  async (/** @type { State } */ state) => {
    try {
      await state.ec2Client.send(
        new DisassociateAddressCommand({
          AssociationId: state.associationId,
        }),
      );
    } catch (caught) {
      if (
        caught instanceof Error &&
        caught.name === "InvalidAssociationID.NotFound"
      ) {
        caught.message = `${caught.message}. Please provide a valid association
        ID.`;
      }
      state.errors.push(caught);
    }
  },
  {
    skipWhen: (/** @type { State } */ state) =>
      !state[confirmDisassociateAddress.name] || !state.associationId,
  },
);

export const maybeReleaseAddress = new ScenarioAction(
  "maybeReleaseAddress",
  async (/** @type { State } */ state) => {
    try {
      await state.ec2Client.send(
```

```
        new ReleaseAddressCommand({
          AllocationId: state.allocationId,
        }),
      );
    } catch (caught) {
      if (
        caught instanceof Error &&
        caught.name === "InvalidAllocationID.NotFound"
      ) {
        caught.message = `${caught.message}. Please provide a valid
AllocationID.`;
      }
      state.errors.push(caught);
    }
  },
  {
    skipWhen: (/** @type { State } */ state) =>
      !state[confirmDisassociateAddress.name] || !state.allocationId,
  },
);

export const confirmTerminateInstance = new ScenarioInput(
  "confirmTerminateInstance",
  "Do you want to terminate the instance?",
  // Don't do anything when an instance was never run.
  {
    skipWhen: (/** @type { State } */ state) => !state.instanceId,
    type: "confirm",
  },
);

export const maybeTerminateInstance = new ScenarioAction(
  "terminateInstance",
  async (/** @type { State } */ state) => {
    try {
      await state.ec2Client.send(
        new TerminateInstancesCommand({
          InstanceIds: [state.instanceId],
        }),
      );
      await waitUntilInstanceTerminated(
        { client: state.ec2Client },
        { InstanceIds: [state.instanceId] },
      );
    }
  }
);
```

```
    } catch (caught) {
      if (caught instanceof Error && caught.name === "TimeoutError") {
        caught.message = `${caught.message}. Try increasing the maxWaitTime in
the waiter.`;
      }

      state.errors.push(caught);
    }
  },
  {
    // Don't do anything when there's no instance to terminate or the
    // use chooses not to terminate.
    skipWhen: (/** @type { State } */ state) =>
      !state.instanceId || !state[confirmTerminateInstance.name],
  },
);

export const deleteTemporaryDirectory = new ScenarioAction(
  "deleteTemporaryDirectory",
  async (/** @type { State } */ state) => {
    try {
      await rm(state.tmpDirectory, { recursive: true });
    } catch (caught) {
      state.errors.push(caught);
    }
  },
);

export const logErrors = new ScenarioOutput(
  "logErrors",
  (/** @type {State}*/ state) => {
    const errorList = state.errors
      .map((err) => ` - ${err.name}: ${err.message}`)
      .join("\n");
    return `Scenario errors found:\n${errorList}`;
  },
  {
    preformatted: true,
    header: true,
    // Don't log errors when there aren't any!
    skipWhen: (/** @type {State} */ state) => state.errors.length === 0,
  },
);
```

- Para API obter detalhes, consulte os tópicos a seguir em AWS SDK for JavaScript APIReferência.
 - [AllocateAddress](#)
 - [AssociateAddress](#)
 - [AuthorizeSecurityGroupIngress](#)
 - [CreateKeyPair](#)
 - [CreateSecurityGroup](#)
 - [DeleteKeyPair](#)
 - [DeleteSecurityGroup](#)
 - [DescribeImages](#)
 - [DescribeInstanceTypes](#)
 - [DescribeInstances](#)
 - [DescribeKeyPairs](#)
 - [DescribeSecurityGroups](#)
 - [DisassociateAddress](#)
 - [ReleaseAddress](#)
 - [RunInstances](#)
 - [StartInstances](#)
 - [StopInstances](#)
 - [TerminateInstances](#)
 - [UnmonitorInstances](#)

Kotlin

SDKpara Kotlin

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
/**
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.waitForInstanceTerminated {
            // 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")
    }
}
}
```

- Para API obter detalhes, consulte os tópicos a seguir AWS SDK para referência do Kotlin API.
 - [AllocateAddress](#)
 - [AssociateAddress](#)
 - [AuthorizeSecurityGroupIngress](#)
 - [CreateKeyPair](#)
 - [CreateSecurityGroup](#)
 - [DeleteKeyPair](#)
 - [DeleteSecurityGroup](#)
 - [DescribeImages](#)
 - [DescribeInstanceTypes](#)
 - [DescribeInstances](#)

- [DescribeKeyPairs](#)
- [DescribeSecurityGroups](#)
- [DisassociateAddress](#)
- [ReleaseAddress](#)
- [RunInstances](#)
- [StartInstances](#)
- [StopInstances](#)
- [TerminateInstances](#)
- [UnmonitorInstances](#)

Python

SDK para Python (Boto3)

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

Execute um cenário interativo em um prompt de comando.

```
class EC2InstanceScenario:
    """
    A scenario that demonstrates how to use Boto3 to manage Amazon EC2 resources.
    Covers creating a key pair, security group, launching an instance,
    associating
    an Elastic IP, and cleaning up resources.
    """

    def __init__(
        self,
        inst_wrapper: EC2InstanceWrapper,
        key_wrapper: KeyPairWrapper,
        sg_wrapper: SecurityGroupWrapper,
        eip_wrapper: ElasticIpWrapper,
        ssm_client: boto3.client,
        remote_exec: bool = False,
```

```

):
    """
    Initializes the EC2InstanceScenario with the necessary AWS service
    wrappers.

    :param inst_wrapper: Wrapper for EC2 instance operations.
    :param key_wrapper: Wrapper for key pair operations.
    :param sg_wrapper: Wrapper for security group operations.
    :param eip_wrapper: Wrapper for Elastic IP operations.
    :param ssm_client: Boto3 client for accessing SSM to retrieve AMIs.
    :param remote_exec: Flag to indicate if the scenario is running in a
    remote execution
                           environment. Defaults to False. If True, the script
    won't prompt
                           for user interaction.
    """
    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
    self.remote_exec = remote_exec

    def create_and_list_key_pairs(self) -> None:
        """
        Creates an RSA key pair for SSH access to the EC2 instance and lists
        available key pairs.
        """
        console.print("***Step 1: Create a Secure Key Pair***", style="bold cyan")
        console.print(
            "Let's create a secure RSA key pair for connecting to your EC2
            instance."
        )
        key_name = f"MyUniqueKeyPair-{uuid.uuid4().hex[:8]}"
        console.print(f"- **Key Pair Name**: {key_name}")

        # Create the key pair and simulate the process with a progress bar.
        with alive_bar(1, title="Creating Key Pair") as bar:
            self.key_wrapper.create(key_name)
            time.sleep(0.4) # Simulate the delay in key creation
            bar()

        console.print(f"- **Private Key Saved to**:"
            {self.key_wrapper.key_file_path}\n")

```

```
# List key pairs (simulated) and show a progress bar.
list_keys = True
if list_keys:
    console.print("- Listing your key pairs...")
    start_time = time.time()
    with alive_bar(100, title="Listing Key Pairs") as bar:
        while time.time() - start_time < 2:
            time.sleep(0.2)
            bar(10)
        self.key_wrapper.list(5)
        if time.time() - start_time > 2:
            console.print(
                "Taking longer than expected! Please wait...",
                style="bold yellow",
            )

def create_security_group(self) -> None:
    """
    Creates a security group that controls access to the EC2 instance and
adds a rule
to allow SSH access from the user's current public IP address.
    """
    console.print("***Step 2: Create a Security Group***", style="bold cyan")
    console.print(
        "Security groups manage access to your instance. Let's create one."
    )
    sg_name = f"MySecurityGroup-{uuid.uuid4().hex[:8]}"
    console.print(f"- **Security Group Name**: {sg_name}")

    # Create the security group and simulate the process with a progress bar.
    with alive_bar(1, title="Creating Security Group") as bar:
        self.sg_wrapper.create(
            sg_name, "Security group for example: get started with
instances."
        )
        time.sleep(0.5)
        bar()

    console.print(f"- **Security Group ID**:
{self.sg_wrapper.security_group}\n")

    # Get the current public IP to set up SSH access.
    ip_response = urllib.request.urlopen("http://checkip.amazonaws.com")
```

```

current_ip_address = ip_response.read().decode("utf-8").strip()
console.print(
    "Let's add a rule to allow SSH only from your current IP address."
)
console.print(f"- **Your Public IP Address**: {current_ip_address}")
console.print("- Automatically adding SSH rule...")

# Update security group rules to allow SSH and simulate with a progress
bar.
with alive_bar(1, title="Updating Security Group Rules") as bar:
    response = self.sg_wrapper.authorize_ingress(current_ip_address)
    time.sleep(0.4)
    if response and response.get("Return"):
        console.print("- **Security Group Rules Updated**.")
    else:
        console.print(
            "- **Error**: Couldn't update security group rules.",
            style="bold red",
        )
    bar()

self.sg_wrapper.describe(self.sg_wrapper.security_group)

def create_instance(self) -> None:
    """
    Launches an EC2 instance using an Amazon Linux 2 AMI and the created key
pair
and security group. Displays instance details and SSH connection
information.
    """
    # Retrieve Amazon Linux 2 AMIs from SSM.
    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"]]
    )
    console.print("\n**Step 3: Launch Your Instance**", style="bold cyan")
    console.print(
        "Let's create an instance from an Amazon Linux 2 AMI. Here are some
options:"
    )

```

```
        image_choice = 0
        console.print(f"- Selected AMI: {amzn2_images[image_choice]
['ImageId']}\n")

        # Display instance types compatible with the selected AMI
        inst_types = self.inst_wrapper.get_instance_types(
            amzn2_images[image_choice]["Architecture"]
        )
        inst_type_choice = 0
        console.print(
            f"- Selected instance type: {inst_types[inst_type_choice]
['InstanceType']}\n"
        )

        console.print("Creating your instance and waiting for it to start..."
with alive_bar(1, title="Creating Instance") as bar:
            self.inst_wrapper.create(
                amzn2_images[image_choice]["ImageId"],
                inst_types[inst_type_choice]["InstanceType"],
                self.key_wrapper.key_pair["KeyName"],
                [self.sg_wrapper.security_group],
            )
            time.sleep(21)
            bar()

        console.print(f"***Success! Your instance is ready:**\n", style="bold
green")
        self.inst_wrapper.display()

        console.print(
            "You can use SSH to connect to your instance. "
            "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) -> None:
        """
        Displays SSH connection information for the user to connect to the EC2
instance.
        Handles the case where the instance does or does not have an associated
public IP address.
```



```
"""
if (
    not self.eip_wrapper.elastic_ips
    or not self.eip_wrapper.elastic_ips[0].allocation_id
):
    if self.inst_wrapper.instances:
        instance = self.inst_wrapper.instances[0]
        instance_id = instance["InstanceId"]

        waiter =
self.inst_wrapper.ec2_client.get_waiter("instance_running")
        console.print(
            "Waiting for the instance to be in a running state with a
public IP...",
            style="bold cyan",
        )

        with alive_bar(1, title="Waiting for Instance to Start") as bar:
            waiter.wait(InstanceIds=[instance_id])
            time.sleep(20)
            bar()

        instance = self.inst_wrapper.ec2_client.describe_instances(
            InstanceIds=[instance_id]
        )["Reservations"][0]["Instances"][0]

        public_ip = instance.get("PublicIpAddress")
        if public_ip:
            console.print(
                "\nTo connect via SSH, open another command prompt and
run the following command:",
                style="bold cyan",
            )
            console.print(
                f"\tssh -i {self.key_wrapper.key_file_path} ec2-
user@{public_ip}"
            )
        else:
            console.print(
                "Instance does not have a public IP address assigned.",
                style="bold red",
            )
    else:
        console.print(
```

```
        "No instance available to retrieve public IP address.",
        style="bold red",
    )
else:
    elastic_ip = self.eip_wrapper.elastic_ips[0]
    elastic_ip_address = elastic_ip.public_ip
    console.print(
        f"\tssh -i {self.key_wrapper.key_file_path} ec2-
user@{elastic_ip_address}"
    )

    if not self.remote_exec:
        console.print("\nOpen a new terminal tab to try the above SSH
command.")
        input("Press Enter to continue...")

def associate_elastic_ip(self) -> None:
    """
    Allocates an Elastic IP address and associates it with the EC2 instance.
    Displays the Elastic IP address and SSH connection information.
    """
    console.print("\n**Step 4: Allocate an Elastic IP Address**", style="bold
cyan")
    console.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."
    )

    with alive_bar(1, title="Allocating Elastic IP") as bar:
        elastic_ip = self.eip_wrapper.allocate()
        time.sleep(0.5)
        bar()

    console.print(
        f"- **Allocated Static Elastic IP Address**: {elastic_ip.public_ip}."
    )

    with alive_bar(1, title="Associating Elastic IP") as bar:
        self.eip_wrapper.associate(
            elastic_ip.allocation_id, self.inst_wrapper.instances[0]
["InstanceId"]
        )
        time.sleep(2)
```

```
        bar()

        console.print(f"- **Associated Elastic IP with Your Instance**.")
        console.print(
            "You can now use SSH to connect to your instance by using the Elastic
IP."
        )
        self._display_ssh_info()

    def stop_and_start_instance(self) -> None:
        """
        Stops and restarts the EC2 instance. Displays instance state and explains
changes
in the public IP address unless an Elastic IP is associated.
        """
        console.print("\n**Step 5: Stop and Start Your Instance**", style="bold
cyan")
        console.print("Let's stop and start your instance to see what changes.")
        console.print("- **Stopping your instance and waiting until it's
stopped...**")

        with alive_bar(1, title="Stopping Instance") as bar:
            self.inst_wrapper.stop()
            time.sleep(360)
            bar()

        console.print("- **Your instance is stopped. Restarting...**")

        with alive_bar(1, title="Starting Instance") as bar:
            self.inst_wrapper.start()
            time.sleep(20)
            bar()

        console.print("**Your instance is running.**", style="bold green")
        self.inst_wrapper.display()

        elastic_ip = (
            self.eip_wrapper.elastic_ips[0] if self.eip_wrapper.elastic_ips else
None
        )

        if elastic_ip is None or elastic_ip.allocation_id is None:
            console.print(
```

```
        "- Note: Every time your instance is restarted, its public IP
address changes."
    )
    else:
        console.print(
            f"Because you have associated an Elastic IP with your instance,
you can \n"
            f"connect by using a consistent IP address after the instance
restarts: {elastic_ip.public_ip}"
        )

        self._display_ssh_info()

    def cleanup(self) -> None:
        """
        Cleans up all the resources created during the scenario, including
disassociating
        and releasing the Elastic IP, terminating the instance, deleting the
security
        group, and deleting the key pair.
        """
        console.print("\nStep 6: Clean Up Resources", style="bold cyan")
        console.print("Cleaning up resources:")

        for elastic_ip in self.eip_wrapper.elastic_ips:
            console.print(f"- Elastic IP: {elastic_ip.public_ip}")

            with alive_bar(1, title="Disassociating Elastic IP") as bar:
                self.eip_wrapper.disassociate(elastic_ip.allocation_id)
                time.sleep(2)
                bar()

            console.print("\t- Disassociated Elastic IP from the Instance")

            with alive_bar(1, title="Releasing Elastic IP") as bar:
                self.eip_wrapper.release(elastic_ip.allocation_id)
                time.sleep(1)
                bar()

            console.print("\t- Released Elastic IP")

        console.print(f"- Instance: {self.inst_wrapper.instances[0]
['InstanceId']}")
```

```
with alive_bar(1, title="Terminating Instance") as bar:
    self.inst_wrapper.terminate()
    time.sleep(380)
    bar()

console.print("\t- **Terminated Instance**")

console.print(f"- **Security Group**: {self.sg_wrapper.security_group}")

with alive_bar(1, title="Deleting Security Group") as bar:
    self.sg_wrapper.delete(self.sg_wrapper.security_group)
    time.sleep(1)
    bar()

console.print("\t- **Deleted Security Group**")

console.print(f"- **Key Pair**: {self.key_wrapper.key_pair['KeyName']}")

with alive_bar(1, title="Deleting Key Pair") as bar:
    self.key_wrapper.delete(self.key_wrapper.key_pair["KeyName"])
    time.sleep(0.4)
    bar()

console.print("\t- **Deleted Key Pair**")

def run_scenario(self) -> None:
    """
    Executes the entire EC2 instance scenario: creates key pairs, security
    groups,
    launches an instance, associates an Elastic IP, and cleans up all
    resources.
    """
    logging.basicConfig(level=logging.INFO, format="%(levelname)s:
%(message)s")

    console.print("-" * 88)
    console.print(
        "Welcome to the Amazon Elastic Compute Cloud (Amazon EC2) get started
with instances demo.",
        style="bold magenta",
    )
    console.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()

console.print("\nThanks for watching!", style="bold green")
console.print("-" * 88)

if __name__ == "__main__":
    try:
        scenario = EC2InstanceScenario(
            EC2InstanceWrapper.from_client(),
            KeyPairWrapper.from_client(),
            SecurityGroupWrapper.from_client(),
            ElasticIpWrapper.from_client(),
            boto3.client("ssm"),
        )
        scenario.run_scenario()
    except Exception:
        logging.exception("Something went wrong with the demo.")
```

Defina uma classe que envolva as ações de pares de chaves.

```
class KeyPairWrapper:
    """
    Encapsulates Amazon Elastic Compute Cloud (Amazon EC2) key pair actions.
    This class provides methods to create, list, and delete EC2 key pairs.
    """

    def __init__(
        self,
        ec2_client: boto3.client,
        key_file_dir: Union[tempfile.TemporaryDirectory, str],
        key_pair: Optional[dict] = None,
    ):
        """
        Initializes the KeyPairWrapper with the specified EC2 client, key file
        directory,
        and an optional key pair.
```

```

        :param ec2_client: A Boto3 Amazon EC2 client. This client provides low-
level
        access to AWS EC2 services.
        :param key_file_dir: The folder where the private key information is
stored.
        This should be a secure folder.
        :param key_pair: A dictionary representing the Boto3 KeyPair object.
        This is a high-level object that wraps key pair actions.
Optional.
    """
    self.ec2_client = ec2_client
    self.key_pair = key_pair
    self.key_file_path: Optional[str] = None
    self.key_file_dir = key_file_dir

    @classmethod
    def from_client(cls) -> "KeyPairWrapper":
        """
        Class method to create an instance of KeyPairWrapper using a new EC2
client
and a temporary directory for storing key files.

        :return: An instance of KeyPairWrapper.
        """
        ec2_client = boto3.client("ec2")
        return cls(ec2_client, tempfile.TemporaryDirectory())

    def create(self, key_name: str) -> dict:
        """
        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 dictionary representing the Boto3 KeyPair object that
represents the newly created key pair.
        :raises ClientError: If there is an error in creating the key pair, for
example, if a key pair with the same name already exists.
        """
        try:

```

```

        response = self.ec2_client.create_key_pair(KeyName=key_name)
        self.key_pair = response
        self.key_file_path = os.path.join(
            self.key_file_dir.name, f"{self.key_pair['KeyName']}.pem"
        )
        with open(self.key_file_path, "w") as key_file:
            key_file.write(self.key_pair["KeyMaterial"])
    except ClientError as err:
        if err.response["Error"]["Code"] == "InvalidKeyPair.Duplicate":
            logger.error(
                f"A key pair called {key_name} already exists. "
                "Please choose a different name for your key pair "
                "or delete the existing key pair before creating."
            )
            raise
        else:
            return self.key_pair

def list(self, limit: Optional[int] = None) -> None:
    """
    Displays a list of key pairs for the current account.

    WARNING: Results are not paginated.

    :param limit: The maximum number of key pairs to list. If not specified,
                  all key pairs will be listed.
    :raises ClientError: If there is an error in listing the key pairs.
    """
    try:
        response = self.ec2_client.describe_key_pairs()
        key_pairs = response.get("KeyPairs", [])

        if limit:
            key_pairs = key_pairs[:limit]

        for key_pair in key_pairs:
            logger.info(
                f"Found {key_pair['KeyType']} key '{key_pair['KeyName']}'
with fingerprint:"
            )
            logger.info(f"\t{key_pair['KeyFingerprint']}")
    except ClientError as err:
        logger.error(f"Failed to list key pairs: {str(err)}")

```



```

        raise

def delete(self, key_name: str) -> bool:
    """
    Deletes a key pair by its name.

    :param key_name: The name of the key pair to delete.
    :return: A boolean indicating whether the deletion was successful.
    :raises ClientError: If there is an error in deleting the key pair, for
example,
                                if the key pair does not exist.
    """
    try:
        self.ec2_client.delete_key_pair(KeyName=key_name)
        logger.info(f"Successfully deleted key pair: {key_name}")
        self.key_pair = None
        return True
    except self.ec2_client.exceptions.ClientError as err:
        logger.error(f"Deletion failed for key pair: {key_name}")
        error_code = err.response["Error"]["Code"]
        if error_code == "InvalidKeyPair.NotFound":
            logger.error(
                f"The key pair '{key_name}' does not exist and cannot be
deleted. "
                "Please verify the key pair name and try again."
            )
        raise

```

Defina uma classe que envolva as ações de grupos de segurança.

```

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

    def __init__(self, ec2_client: boto3.client, security_group: Optional[str] =
None):
        """

```

```

        Initializes the SecurityGroupWrapper with an EC2 client and an optional
        security group ID.

        :param ec2_client: A Boto3 Amazon EC2 client. This client provides low-
level
                            access to AWS EC2 services.
        :param security_group: The ID of a security group to manage. This is a
high-level identifier
                            that represents the security group.
        """
        self.ec2_client = ec2_client
        self.security_group = security_group

    @classmethod
    def from_client(cls) -> "SecurityGroupWrapper":
        """
        Creates a SecurityGroupWrapper instance with a default EC2 client.

        :return: An instance of SecurityGroupWrapper initialized with the default
EC2 client.
        """
        ec2_client = boto3.client("ec2")
        return cls(ec2_client)

    def create(self, group_name: str, group_description: str) -> str:
        """
        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: The ID of the newly created security group.
        :raise Handles AWS SDK service-level ClientError, with special handling
for ResourceAlreadyExists
        """
        try:
            response = self.ec2_client.create_security_group(
                GroupName=group_name, Description=group_description
            )
            self.security_group = response["GroupId"]
        except ClientError as err:
            if err.response["Error"]["Code"] == "ResourceAlreadyExists":

```

```
        logger.error(
            f"Security group '{group_name}' already exists. Please choose
a different name."
        )
        raise
    else:
        return self.security_group

def authorize_ingress(self, ssh_ingress_ip: str) -> Optional[Dict[str, Any]]:
    """
    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, or
None if no security group is set.
    :raise: Handles AWS SDK service-level ClientError, with special handling
for ResourceAlreadyExists
    """
    if self.security_group is None:
        logger.info("No security group to update.")
        return None

    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.ec2_client.authorize_security_group_ingress(
            GroupId=self.security_group, IpPermissions=ip_permissions
        )
    except ClientError as err:
        if err.response["Error"]["Code"] == "InvalidPermission.Duplicate":
            logger.error(
```

```

        f"The SSH ingress rule for IP {ssh_ingress_ip} already
exists"
        f"in security group '{self.security_group}'."
    )
    raise
else:
    return response

def describe(self, security_group_id: Optional[str] = None) -> bool:
    """
    Displays information about the specified security group or all security
    groups if no ID is provided.

    :param security_group_id: The ID of the security group to describe.
        If None, an open search is performed to
    describe all security groups.
    :returns: True if the description is successful.
    :raises ClientError: If there is an error describing the security
    group(s), such as an invalid security group ID.
    """
    try:
        paginator = self.ec2_client.get_paginator("describe_security_groups")

        if security_group_id is None:
            # If no ID is provided, return all security groups.
            page_iterator = paginator.paginate()
        else:
            page_iterator = paginator.paginate(GroupIds=[security_group_id])

        for page in page_iterator:
            for security_group in page["SecurityGroups"]:
                print(f"Security group: {security_group['GroupName']}")
                print(f"\tID: {security_group['GroupId']}")
                print(f"\tVPC: {security_group['VpcId']}")
                if security_group["IpPermissions"]:
                    print("Inbound permissions:")
                    pp(security_group["IpPermissions"])

        return True
    except ClientError as err:
        logger.error("Failed to describe security group(s).")
        if err.response["Error"]["Code"] == "InvalidGroup.NotFound":
            logger.error(

```

```
        f"Security group {security_group_id} does not exist "
        f"because the specified security group ID was not found."
    )
    raise

def delete(self, security_group_id: str) -> bool:
    """
    Deletes the specified security group.

    :param security_group_id: The ID of the security group to delete.
    Required.

    :returns: True if the deletion is successful.
    :raises ClientError: If the security group cannot be deleted due to an
    AWS service error.
    """
    try:
        self.ec2_client.delete_security_group(GroupId=security_group_id)
        logger.info(f"Successfully deleted security group
        '{security_group_id}'")
        return True
    except ClientError as err:
        logger.error(f"Deletion failed for security group
        '{security_group_id}'")
        error_code = err.response["Error"]["Code"]

        if error_code == "InvalidGroup.NotFound":
            logger.error(
                f"Security group '{security_group_id}' cannot be deleted
                because it does not exist."
            )
        elif error_code == "DependencyViolation":
            logger.error(
                f"Security group '{security_group_id}' cannot be deleted
                because it is still in use."
                " Verify that it is:"
                "\n\t- Detached from resources"
                "\n\t- Removed from references in other groups"
                "\n\t- Removed from VPC's as a default group"
            )
        raise
```

Defina uma classe que envolva as ações de instâncias.

```
class EC2InstanceWrapper:
    """Encapsulates Amazon Elastic Compute Cloud (Amazon EC2) instance actions
    using the client interface."""

    def __init__(
        self, ec2_client: Any, instances: Optional[List[Dict[str, Any]]] = None
    ) -> None:
        """
        Initializes the EC2InstanceWrapper with an EC2 client and optional
        instances.

        :param ec2_client: A Boto3 Amazon EC2 client. This client provides low-
        level
                           access to AWS EC2 services.
        :param instances: A list of dictionaries representing Boto3 Instance
        objects. These are high-level objects that
                           wrap instance actions.
        """
        self.ec2_client = ec2_client
        self.instances = instances or []

    @classmethod
    def from_client(cls) -> "EC2InstanceWrapper":
        """
        Creates an EC2InstanceWrapper instance with a default EC2 client.

        :return: An instance of EC2InstanceWrapper initialized with the default
        EC2 client.
        """
        ec2_client = boto3.client("ec2")
        return cls(ec2_client)

    def create(
        self,
        image_id: str,
        instance_type: str,
        key_pair_name: str,
```

```

        security_group_ids: Optional[List[str]] = None,
    ) -> List[Dict[str, Any]]:
        """
        Creates a new EC2 instance in the default VPC of the current account.

        The instance starts immediately after it is created.

        :param image_id: The ID of the Amazon Machine Image (AMI) to use for the
instance.
        :param instance_type: The type of instance to create, such as 't2.micro'.
        :param key_pair_name: The name of the key pair to use for SSH access.
        :param security_group_ids: A list of security group IDs to associate with
the instance.
                                If not specified, the default security group
of the VPC is used.
        :return: A list of dictionaries representing Boto3 Instance objects
representing the newly created instances.
        """
        try:
            instance_params = {
                "ImageId": image_id,
                "InstanceType": instance_type,
                "KeyName": key_pair_name,
            }
            if security_group_ids is not None:
                instance_params["SecurityGroupIds"] = security_group_ids

            response = self.ec2_client.run_instances(
                **instance_params, MinCount=1, MaxCount=1
            )
            instance = response["Instances"][0]
            self.instances.append(instance)
            waiter = self.ec2_client.get_waiter("instance_running")
            waiter.wait(InstanceIds=[instance["InstanceId"]])
        except ClientError as err:
            params_str = "\n\t".join(
                f"{key}: {value}" for key, value in instance_params.items()
            )
            logger.error(
                f"Failed to complete instance creation request.\nRequest details:
{params_str}"
            )
            error_code = err.response["Error"]["Code"]
            if error_code == "InstanceLimitExceeded":

```

```
        logger.error(
            (
                f"Insufficient capacity for instance type
'{instance_type}'. "
                "Terminate unused instances or contact AWS Support for a
limit increase."
            )
        )
        if error_code == "InsufficientInstanceCapacity":
            logger.error(
                (
                    f"Insufficient capacity for instance type
'{instance_type}'. "
                    "Select a different instance type or launch in a
different availability zone."
                )
            )
            raise
        return self.instances

def display(self, state_filter: Optional[str] = "running") -> None:
    """
    Displays information about instances, filtering by the specified state.

    :param state_filter: The instance state to include in the output. Only
instances in this state
                        will be displayed. Default is 'running'. Example
states: 'running', 'stopped'.
    """
    if not self.instances:
        logger.info("No instances to display.")
        return

    instance_ids = [instance["InstanceId"] for instance in self.instances]
    paginator = self.ec2_client.get_paginator("describe_instances")
    page_iterator = paginator.paginate(InstanceIds=instance_ids)

    try:
        for page in page_iterator:
            for reservation in page["Reservations"]:
                for instance in reservation["Instances"]:
                    instance_state = instance["State"]["Name"]
```



```

        # Apply the state filter (default is 'running')
        if state_filter and instance_state != state_filter:
            continue # Skip this instance if it doesn't match
the filter

        # Create a formatted string with instance details
        instance_info = (
            f"• ID: {instance['InstanceId']}\n"
            f"• Image ID: {instance['ImageId']}\n"
            f"• Instance type: {instance['InstanceType']}\n"
            f"• Key name: {instance['KeyName']}\n"
            f"• VPC ID: {instance['VpcId']}\n"
            f"• Public IP: {instance.get('PublicIpAddress', 'N/A')}\n"
            f"• State: {instance_state}"
        )
        print(instance_info)

    except ClientError as err:
        logger.error(
            f"Failed to display instance(s). : {' '.join(map(str,
instance_ids))}"
        )
        error_code = err.response["Error"]["Code"]
        if error_code == "InvalidInstanceID.NotFound":
            logger.error(
                "One or more instance IDs do not exist. "
                "Please verify the instance IDs and try again."
            )
            raise

def terminate(self) -> None:
    """
    Terminates instances and waits for them to reach the terminated state.
    """
    if not self.instances:
        logger.info("No instances to terminate.")
        return

    instance_ids = [instance["InstanceId"] for instance in self.instances]
    try:
        self.ec2_client.terminate_instances(InstanceIds=instance_ids)
        waiter = self.ec2_client.get_waiter("instance_terminated")

```

```

        waiter.wait(InstanceIds=instance_ids)
        self.instances.clear()
        for instance_id in instance_ids:
            print(f"• Instance ID: {instance_id}\n" f"• Action: Terminated")

except ClientError as err:
    logger.error(
        f"Failed instance termination details:\n\t{str(self.instances)}"
    )
    error_code = err.response["Error"]["Code"]
    if error_code == "InvalidInstanceID.NotFound":
        logger.error(
            "One or more instance IDs do not exist. "
            "Please verify the instance IDs and try again."
        )
    raise

def start(self) -> Optional[Dict[str, Any]]:
    """
    Starts instances and waits for them to be in a running state.

    :return: The response to the start request.
    """
    if not self.instances:
        logger.info("No instances to start.")
        return None

    instance_ids = [instance["InstanceId"] for instance in self.instances]
    try:
        start_response =
self.ec2_client.start_instances(InstanceIds=instance_ids)
        waiter = self.ec2_client.get_waiter("instance_running")
        waiter.wait(InstanceIds=instance_ids)
        return start_response
    except ClientError as err:
        logger.error(
            f"Failed to start instance(s): {','.join(map(str,
instance_ids))}"
        )
        error_code = err.response["Error"]["Code"]
        if error_code == "IncorrectInstanceState":
            logger.error(

```

```

        "Couldn't start instance(s) because they are in an incorrect
state. "
        "Ensure the instances are in a stopped state before starting
them."
    )
    raise

def stop(self) -> Optional[Dict[str, Any]]:
    """
    Stops instances and waits for them to be in a stopped state.

    :return: The response to the stop request, or None if there are no
instances to stop.
    """
    if not self.instances:
        logger.info("No instances to stop.")
        return None

    instance_ids = [instance["InstanceId"] for instance in self.instances]
    try:
        # Attempt to stop the instances
        stop_response =
self.ec2_client.stop_instances(InstanceIds=instance_ids)
        waiter = self.ec2_client.get_waiter("instance_stopped")
        waiter.wait(InstanceIds=instance_ids)
    except ClientError as err:
        logger.error(
            f"Failed to stop instance(s): {' '.join(map(str, instance_ids))}"
        )
        error_code = err.response["Error"]["Code"]
        if error_code == "IncorrectInstanceState":
            logger.error(
                "Couldn't stop instance(s) because they are in an incorrect
state. "
                "Ensure the instances are in a running state before stopping
them."
            )
            raise
    return stop_response

def get_images(self, image_ids: List[str]) -> List[Dict[str, Any]]:
    """

```

```

Gets information about Amazon Machine Images (AMIs) from a list of AMI
IDs.

:param image_ids: The list of AMI IDs to look up.
:return: A list of dictionaries representing the requested AMIs.
"""
try:
    response = self.ec2_client.describe_images(ImageIds=image_ids)
    images = response["Images"]
except ClientError as err:
    logger.error(f"Failed to stop AMI(s): {','.join(map(str,
image_ids))}")
    error_code = err.response["Error"]["Code"]
    if error_code == "InvalidAMIID.NotFound":
        logger.error("One or more of the AMI IDs does not exist.")
    raise
return images

def get_instance_types(
    self, architecture: str = "x86_64", sizes: List[str] = ["*.micro",
"*.small"]
) -> List[Dict[str, Any]]:
    """
    Gets instance types that support the specified architecture and size.
    See https://docs.aws.amazon.com/AWSEC2/latest/APIReference/
API\_DescribeInstanceTypes.html
    for a list of allowable parameters.

    :param architecture: The architecture supported by instance types.
    Default: 'x86_64'.
    :param sizes: The size of instance types. Default: '*.micro', '*.small',
    :return: A list of dictionaries representing instance types that support
    the specified architecture and size.
    """
    try:
        inst_types = []
        paginator = self.ec2_client.get_paginator("describe_instance_types")
        for page in paginator.paginate(
            Filters=[
                {
                    "Name": "processor-info.supported-architecture",
                    "Values": [architecture],
                },
            ],

```

```

        {"Name": "instance-type", "Values": sizes},
    ]
):
    inst_types += page["InstanceTypes"]
except ClientError as err:
    logger.error(
        f"Failed to get instance types: {architecture},
{'.'.join(map(str, sizes))}"
    )
    error_code = err.response["Error"]["Code"]
    if error_code == "InvalidParameterValue":
        logger.error(
            "Parameters are invalid. "
            "Ensure architecture and size strings conform to
DescribeInstanceTypes API reference."
        )
        raise
    else:
        return inst_types

```

Defina uma classe que envolva as ações de IP elástico.

```

class ElasticIpWrapper:
    """Encapsulates Amazon Elastic Compute Cloud (Amazon EC2) Elastic IP address
    actions using the client interface."""

    class ElasticIp:
        """Represents an Elastic IP and its associated instance."""

        def __init__(
            self, allocation_id: str, public_ip: str, instance_id: Optional[str]
            = None
        ) -> None:
            """
            Initializes the ElasticIp object.

            :param allocation_id: The allocation ID of the Elastic IP.
            :param public_ip: The public IP address of the Elastic IP.
            :param instance_id: The ID of the associated EC2 instance, if any.

```

```

        """
        self.allocation_id = allocation_id
        self.public_ip = public_ip
        self.instance_id = instance_id

def __init__(self, ec2_client: Any) -> None:
    """
    Initializes the ElasticIpWrapper with an EC2 client.

    :param ec2_client: A Boto3 Amazon EC2 client. This client provides low-
level
                        access to AWS EC2 services.
    """
    self.ec2_client = ec2_client
    self.elastic_ips: List[ElasticIpWrapper.ElasticIp] = []

    @classmethod
    def from_client(cls) -> "ElasticIpWrapper":
        """
        Creates an ElasticIpWrapper instance with a default EC2 client.

        :return: An instance of ElasticIpWrapper initialized with the default EC2
client.
        """
        ec2_client = boto3.client("ec2")
        return cls(ec2_client)

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

        :return: The ElasticIp object for the newly created Elastic IP address.
        :raises ClientError: If the allocation fails, such as reaching the
maximum limit of Elastic IPs.
        """
        try:
            response = self.ec2_client.allocate_address(Domain="vpc")
            elastic_ip = self.ElasticIp(
                allocation_id=response["AllocationId"],
public_ip=response["PublicIp"]

```

```
    )
    self.elastic_ips.append(elastic_ip)
except ClientError as err:
    if err.response["Error"]["Code"] == "AddressLimitExceeded":
        logger.error(
            "Max IP's reached. Release unused addresses or contact AWS
Support for an increase."
        )
        raise err
return elastic_ip

def associate(
    self, allocation_id: str, instance_id: str
) -> Union[Dict[str, Any], None]:
    """
    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 allocation_id: The allocation ID of the Elastic IP.
    :param instance_id: The ID of the Amazon EC2 instance.
    :return: A response that contains the ID of the association, or None if
no Elastic IP is found.
    :raises ClientError: If the association fails, such as when the instance
ID is not found.
    """
    elastic_ip = self.get_elastic_ip_by_allocation(self.elastic_ips,
allocation_id)
    if elastic_ip is None:
        logger.info(f"No Elastic IP found with allocation ID
{allocation_id}.")
        return None

    try:
        response = self.ec2_client.associate_address(
            AllocationId=allocation_id, InstanceId=instance_id
        )
        elastic_ip.instance_id = (
            instance_id # Track the instance associated with this Elastic
IP.
        )
```

```
except ClientError as err:
    if err.response["Error"]["Code"] == "InvalidInstanceID.NotFound":
        logger.error(
            f"Failed to associate Elastic IP {allocation_id} with
{instance_id} "
            "because the specified instance ID does not exist or has not
propagated fully. "
            "Verify the instance ID and try again, or wait a few moments
before attempting to "
            "associate the Elastic IP address."
        )
        raise
    return response

def disassociate(self, allocation_id: str) -> None:
    """
    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.

    :param allocation_id: The allocation ID of the Elastic IP to
disassociate.
    :raises ClientError: If the disassociation fails, such as when the
association ID is not found.
    """
    elastic_ip = self.get_elastic_ip_by_allocation(self.elastic_ips,
allocation_id)
    if elastic_ip is None or elastic_ip.instance_id is None:
        logger.info(
            f"No association found for Elastic IP with allocation ID
{allocation_id}."
        )
        return

    try:
        # Retrieve the association ID before disassociating
        response =
self.ec2_client.describe_addresses(AllocationIds=[allocation_id])
        association_id = response["Addresses"][0].get("AssociationId")

        if association_id:

self.ec2_client.disassociate_address(AssociationId=association_id)
```



```
        elastic_ip.instance_id = None # Remove the instance association
    else:
        logger.info(
            f"No Association ID found for Elastic IP with allocation ID
{allocation_id}."
        )

    except ClientError as err:
        if err.response["Error"]["Code"] == "InvalidAssociationID.NotFound":
            logger.error(
                f"Failed to disassociate Elastic IP {allocation_id} "
                "because the specified association ID for the Elastic IP
address was not found. "
                "Verify the association ID and ensure the Elastic IP is
currently associated with a "
                "resource before attempting to disassociate it."
            )
            raise

def release(self, allocation_id: str) -> None:
    """
    Releases an Elastic IP address. After the Elastic IP address is released,
    it can no longer be used.

    :param allocation_id: The allocation ID of the Elastic IP to release.
    :raises ClientError: If the release fails, such as when the Elastic IP
address is not found.
    """
    elastic_ip = self.get_elastic_ip_by_allocation(self.elastic_ips,
allocation_id)
    if elastic_ip is None:
        logger.info(f"No Elastic IP found with allocation ID
{allocation_id}.")
        return

    try:
        self.ec2_client.release_address(AllocationId=allocation_id)
        self.elastic_ips.remove(elastic_ip) # Remove the Elastic IP from the
list
    except ClientError as err:
        if err.response["Error"]["Code"] == "InvalidAddress.NotFound":
            logger.error(
                f"Failed to release Elastic IP address {allocation_id} "
```

```

        "because it could not be found. Verify the Elastic IP address
"
        "and ensure it is allocated to your account in the correct
region "
        "before attempting to release it."
    )
    raise

    @staticmethod
    def get_elastic_ip_by_allocation(
        elastic_ips: List["ElasticIpWrapper.ElasticIp"], allocation_id: str
    ) -> Optional["ElasticIpWrapper.ElasticIp"]:
        """
        Retrieves an Elastic IP object by its allocation ID from a given list of
        Elastic IPs.

        :param elastic_ips: A list of ElasticIp objects.
        :param allocation_id: The allocation ID of the Elastic IP to retrieve.
        :return: The ElasticIp object associated with the allocation ID, or None
        if not found.
        """
        return next(
            (ip for ip in elastic_ips if ip.allocation_id == allocation_id), None
        )

```

- Para API obter detalhes, consulte os tópicos a seguir em AWS SDKfor Python (Boto3) Reference. API
 - [AllocateAddress](#)
 - [AssociateAddress](#)
 - [AuthorizeSecurityGroupIngress](#)
 - [CreateKeyPair](#)
 - [CreateSecurityGroup](#)
 - [DeleteKeyPair](#)
 - [DeleteSecurityGroup](#)
 - [DescribeImages](#)

- [DescribeInstanceTypes](#)
- [DescribeInstances](#)
- [DescribeKeyPairs](#)
- [DescribeSecurityGroups](#)
- [DisassociateAddress](#)
- [ReleaseAddress](#)
- [RunInstances](#)
- [StartInstances](#)
- [StopInstances](#)
- [TerminateInstances](#)
- [UnmonitorInstances](#)

Rust

SDK para Rust

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

A `EC2InstanceScenario` implementação contém lógica para executar o exemplo como um todo.

```
//! Scenario that uses the AWS SDK for Rust (the SDK) with Amazon Elastic Compute
    Cloud
//! (Amazon EC2) to do the following:
//!
//! * Create a key pair that is used to secure SSH communication between your
    computer and
//!   an EC2 instance.
//! * Create a security group that acts as a virtual firewall for your EC2
    instances to
//!   control incoming and outgoing traffic.
//! * Find an Amazon Machine Image (AMI) and a compatible instance type.
```

```
#!/ * Create an instance that is created from the instance type and AMI you
select, and
#!/ is configured to use the security group and key pair created in this
example.
#!/ * Stop and restart the instance.
#!/ * Create an Elastic IP address and associate it as a consistent IP address
for your instance.
#!/ * Connect to your instance with SSH, using both its public IP address and
your Elastic IP
#!/ address.
#!/ * Clean up all of the resources created by this example.

use std::net::Ipv4Addr;

use crate::{
    ec2::{EC2Error, EC2},
    getting_started::{key_pair::KeyPairManager, util::Util},
    ssm::SSM,
};
use aws_sdk_ssm::types::Parameter;

use super::{
    elastic_ip::ElasticIpManager, instance::InstanceManager,
    security_group::SecurityGroupManager,
    util::ScenarioImage,
};

pub struct Ec2InstanceScenario {
    ec2: EC2,
    ssm: SSM,
    util: Util,
    key_pair_manager: KeyPairManager,
    security_group_manager: SecurityGroupManager,
    instance_manager: InstanceManager,
    elastic_ip_manager: ElasticIpManager,
}

impl Ec2InstanceScenario {
    pub fn new(ec2: EC2, ssm: SSM, util: Util) -> Self {
        Ec2InstanceScenario {
            ec2,
            ssm,
            util,
            key_pair_manager: Default::default(),
        }
    }
}
```

```

        security_group_manager: Default::default(),
        instance_manager: Default::default(),
        elastic_ip_manager: Default::default(),
    }
}

pub async fn run(&mut self) -> Result<(), EC2Error> {
    self.create_and_list_key_pairs().await?;
    self.create_security_group().await?;
    self.create_instance().await?;
    self.stop_and_start_instance().await?;
    self.associate_elastic_ip().await?;
    self.stop_and_start_instance().await?;
    Ok(())
}

/// 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. Optionally, lists the first five key pairs for the current account.
pub async fn create_and_list_key_pairs(&mut self) -> Result<(), EC2Error> {
    println!( "Let's create an RSA key pair that you can be use to securely
connect to your EC2 instance.");

    let key_name = self.util.prompt_key_name()?;

    self.key_pair_manager
        .create(&self.ec2, &self.util, key_name)
        .await?;

    println!(
        "Created a key pair {} and saved the private key to {:?}.",
        self.key_pair_manager
            .key_pair()
            .key_name()
            .ok_or_else(|| EC2Error::new("No key name after creating key"))?,
        self.key_pair_manager
            .key_file_path()
            .ok_or_else(|| EC2Error::new("No key file after creating key"))?
    );

    if self.util.should_list_key_pairs()? {
        for pair in self.key_pair_manager.list(&self.ec2).await? {

```

```

        println!(
            "Found {:?} key {} with fingerprint:\t{:?}",
            pair.key_type(),
            pair.key_name().unwrap_or("Unknown"),
            pair.key_fingerprint()
        );
    }
}

Ok(())
}

/// 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.
pub async fn create_security_group(&mut self) -> Result<(), EC2Error> {
    println!("Let's create a security group to manage access to your
instance.");
    let group_name = self.util.prompt_security_group_name()?;

    self.security_group_manager
        .create(
            &self.ec2,
            &group_name,
            "Security group for example: get started with instances.",
        )
        .await?;

    println!(
        "Created security group {} in your default VPC {}.",
        self.security_group_manager.group_name(),
        self.security_group_manager
            .vpc_id()
    );
}

```

```

        .unwrap_or("(unknown vpc)")
    );

    let check_ip = self.util.do_get("https://checkip.amazonaws.com").await?;
    let current_ip_address: Ipv4Addr = check_ip.trim().parse().map_err(|e| {
        EC2Error::new(format!(
            "Failed to convert response {} to IP Address: {e:?}",
            check_ip
        ))
    })?;

    println!("Your public IP address seems to be {current_ip_address}");
    if self.util.should_add_to_security_group() {
        match self
            .security_group_manager
            .authorize_ingress(&self.ec2, current_ip_address)
            .await
        {
            Ok(_) => println!("Security group rules updated"),
            Err(err) => eprintln!("Couldn't update security group rules:
{err:?}"),
        }
    }
    println!("{}", self.security_group_manager);

    Ok(())
}

/// 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.
pub async fn create_instance(&mut self) -> Result<(), EC2Error> {
    let ami = self.find_image().await?;

```

```
let instance_types = self
    .ec2
    .list_instance_types(&ami.0)
    .await
    .map_err(|e| e.add_message("Could not find instance types"))?;
println!(
    "There are several instance types that support the {} architecture of
the image.",
    ami.0
    .architecture
    .as_ref()
    .ok_or_else(|| EC2Error::new(format!("Missing architecture in
{:?}", ami.0)))?
);
let instance_type = self.util.select_instance_type(instance_types)?;

println!("Creating your instance and waiting for it to start...");
self.instance_manager
    .create(
        &self.ec2,
        ami.0
            .image_id()
            .ok_or_else(|| EC2Error::new("Could not find image ID"))?,
        instance_type,
        self.key_pair_manager.key_pair(),
        self.security_group_manager
            .security_group()
            .map(|sg| vec![sg])
            .ok_or_else(|| EC2Error::new("Could not find security
group"))?,
    )
    .await
    .map_err(|e| e.add_message("Scenario failed to create instance"))?;

while let Err(err) = self
    .ec2
    .wait_for_instance_ready(self.instance_manager.instance_id(), None)
    .await
{
    println!("{}", err);
    if !self.util.should_continue_waiting() {
        return Err(err);
    }
}
```



```

    }

    println!("Your instance is ready:\n{}", self.instance_manager);

    self.display_ssh_info();

    Ok(())
}

async fn find_image(&mut self) -> Result<ScenarioImage, EC2Error> {
    let params: Vec<Parameter> = self
        .ssm
        .list_path("/aws/service/ami-amazon-linux-latest")
        .await
        .map_err(|e| e.add_message("Could not find parameters for available
images"))?
        .into_iter()
        .filter(|param| param.name().is_some_and(|name|
name.contains("amzn2")))
        .collect();
    let amzn2_images: Vec<ScenarioImage> = self
        .ec2
        .list_images(params)
        .await
        .map_err(|e| e.add_message("Could not find images"))?
        .into_iter()
        .map(ScenarioImage::from)
        .collect();
    println!("We will now create an instance from an Amazon Linux 2 AMI");
    let ami = self.util.select_scenario_image(amzn2_images)?;
    Ok(ami)
}

// 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.
pub async fn stop_and_start_instance(&self) -> Result<(), EC2Error> {
    println!("Let's stop and start your instance to see what changes.");
    println!("Stopping your instance and waiting until it's stopped...");
}

```

```
self.instance_manager.stop(&self.ec2).await?;
println!("Your instance is stopped. Restarting...");
self.instance_manager.start(&self.ec2).await?;
println!("Your instance is running.");
println!("{}", self.instance_manager);
if self.elastic_ip_manager.public_ip() == "0.0.0.0" {
    println!("Every time your instance is restarted, its public IP
address changes.");
} else {
    println!(
        "Because you have associated an Elastic IP with your instance,
you can connect by using a consistent IP address after the instance restarts."
    );
}
self.display_ssh_info();
Ok(())
}

/// 1. Allocates an Elastic IP address and associates it with the instance.
/// 2. Displays an SSH connection string that uses the Elastic IP address.
async fn associate_elastic_ip(&mut self) -> Result<(), EC2Error> {
    self.elastic_ip_manager.allocate(&self.ec2).await?;
    println!(
        "Allocated static Elastic IP address: {}",
        self.elastic_ip_manager.public_ip()
    );

    self.elastic_ip_manager
        .associate(&self.ec2, self.instance_manager.instance_id())
        .await?;
    println!("Associated your Elastic IP with your instance.");
    println!("You can now use SSH to connect to your instance by using the
Elastic IP.");
    self.display_ssh_info();
    Ok(())
}

/// Displays an SSH connection string that can be used to connect to a
running
/// instance.
fn display_ssh_info(&self) {
    let ip_addr = if self.elastic_ip_manager.has_allocation() {
        self.elastic_ip_manager.public_ip()
    } else {
```

```

        self.instance_manager.instance_ip()
    };
    let key_file_path = self.key_pair_manager.key_file_path().unwrap();
    println!("To connect, open another command prompt and run the following
command:");
    println!("\nssh -i {} ec2-user@{ip_addr}\n", key_file_path.display());
    let _ = self.util.enter_to_continue();
}

/// 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.
pub async fn clean_up(self) {
    println!("Let's clean everything up. This example created these
resources:");
    println!(
        "\tKey pair: {}",
        self.key_pair_manager
            .key_pair()
            .key_name()
            .unwrap_or("(unknown key pair)")
    );
    println!(
        "\tSecurity group: {}",
        self.security_group_manager.group_name()
    );
    println!(
        "\tInstance: {}",
        self.instance_manager.instance_display_name()
    );
    if self.util.should_clean_resources() {
        if let Err(err) = self.elastic_ip_manager.remove(&self.ec2).await {
            eprintln!("{err}")
        }
        if let Err(err) = self.instance_manager.delete(&self.ec2).await {
            eprintln!("{err}")
        }
        if let Err(err) = self.security_group_manager.delete(&self.ec2).await
{
            eprintln!("{err}");
        }
        if let Err(err) = self.key_pair_manager.delete(&self.ec2,
&self.util).await {

```

```

        eprintln!("{err}");
    }
} else {
    println!("Ok, not cleaning up any resources!");
}
}
}

pub async fn run(mut scenario: Ec2InstanceScenario) {
    println!
    ("-----");
    println!(
        "Welcome to the Amazon Elastic Compute Cloud (Amazon EC2) get started
        with instances demo."
    );
    println!
    ("-----");

    if let Err(err) = scenario.run().await {
        eprintln!("There was an error running the scenario: {err}")
    }

    println!
    ("-----");

    scenario.clean_up().await;

    println!("Thanks for running!");
    println!
    ("-----");
}
}

```

A `EC2Impl` estrutura serve como um ponto automático para testes e suas funções envolvem as EC2 SDK chamadas.

```

use std::{net::Ipv4Addr, time::Duration};

use aws_sdk_ec2::{
    client::Waiters,
    error::ProvideErrorMetadata,
    operation::{

```

```

        allocate_address::AllocateAddressOutput,
    associate_address::AssociateAddressOutput,
    },
    types::{
        DomainType, Filter, Image, Instance, InstanceType, IpPermission, IpRange,
        KeyPairInfo,
        SecurityGroup, Tag,
    },
    Client as EC2Client,
};
use aws_sdk_ssm::types::Parameter;
use aws_smithy_runtime_api::client::waiters::error::WaiterError;

#[cfg(test)]
use mockall::automock;

#[cfg(not(test))]
pub use EC2Impl as EC2;

#[cfg(test)]
pub use MockEC2Impl as EC2;

#[derive(Clone)]
pub struct EC2Impl {
    pub client: EC2Client,
}

#[cfg_attr(test, automock)]
impl EC2Impl {
    pub fn new(client: EC2Client) -> Self {
        EC2Impl { client }
    }

    pub async fn create_key_pair(&self, name: String) -> Result<(KeyPairInfo,
String), EC2Error> {
        tracing::info!("Creating key pair {name}");
        let output = self.client.create_key_pair().key_name(name).send().await?;
        let info = KeyPairInfo::builder()
            .set_key_name(output.key_name)
            .set_key_fingerprint(output.key_fingerprint)
            .set_key_pair_id(output.key_pair_id)
            .build();
        let material = output
            .key_material

```

```
        .ok_or_else(|| EC2Error::new("Create Key Pair has no key
material"))?);
        Ok((info, material))
    }

    pub async fn list_key_pair(&self) -> Result<Vec<KeyPairInfo>, EC2Error> {
        let output = self.client.describe_key_pairs().send().await?;
        Ok(output.key_pairs.unwrap_or_default())
    }

    pub async fn delete_key_pair(&self, key_pair_id: &str) -> Result<(),
EC2Error> {
        let key_pair_id: String = key_pair_id.into();
        tracing::info!("Deleting key pair {key_pair_id}");
        self.client
            .delete_key_pair()
            .key_pair_id(key_pair_id)
            .send()
            .await?;
        Ok(())
    }

    pub async fn create_security_group(
        &self,
        name: &str,
        description: &str,
    ) -> Result<SecurityGroup, EC2Error> {
        tracing::info!("Creating security group {name}");
        let create_output = self
            .client
            .create_security_group()
            .group_name(name)
            .description(description)
            .send()
            .await
            .map_err(EC2Error::from)?;

        let group_id = create_output
            .group_id
            .ok_or_else(|| EC2Error::new("Missing security group id after
creation"))?;

        let group = self
            .describe_security_group(&group_id)
```

```

        .await?
        .ok_or_else(|| {
            EC2Error::new(format!("Could not find security group with id
{group_id}"))
        })?;

        tracing::info!("Created security group {name} as {group_id}");

        Ok(group)
    }

    /// Find a single security group, by ID. Returns Err if multiple groups are
    found.
    pub async fn describe_security_group(
        &self,
        group_id: &str,
    ) -> Result<Option<SecurityGroup>, EC2Error> {
        let group_id: String = group_id.into();
        let describe_output = self
            .client
            .describe_security_groups()
            .group_ids(&group_id)
            .send()
            .await?;

        let mut groups = describe_output.security_groups.unwrap_or_default();

        match groups.len() {
            0 => Ok(None),
            1 => Ok(Some(groups.remove(0))),
            _ => Err(EC2Error::new(format!(
                "Expected single group for {group_id}"
            ))),
        }
    }
}

/// Add an ingress rule to a security group explicitly allowing IPv4 address
/// as {ip}/32 over TCP port 22.
pub async fn authorize_security_group_ssh_ingress(
    &self,
    group_id: &str,
    ingress_ips: Vec<Ipv4Addr>,
) -> Result<(), EC2Error> {
    tracing::info!("Authorizing ingress for security group {group_id}");

```

```

        self.client
            .authorize_security_group_ingress()
            .group_id(group_id)
            .set_ip_permissions(Some(
                ingress_ips
                    .into_iter()
                    .map(|ip| {
                        IpPermission::builder()
                            .ip_protocol("tcp")
                            .from_port(22)
                            .to_port(22)
                            .ip_ranges(IpRange::builder().cidr_ip(format!
                                ("{{ip}}/32")).build())
                            .build()
                    })
                .collect(),
            ))
            .send()
            .await?;
        Ok(())
    }

    pub async fn delete_security_group(&self, group_id: &str) -> Result<(),
    EC2Error> {
        tracing::info!("Deleting security group {group_id}");
        self.client
            .delete_security_group()
            .group_id(group_id)
            .send()
            .await?;
        Ok(())
    }

    pub async fn list_images(&self, ids: Vec<Parameter>) -> Result<Vec<Image>,
    EC2Error> {
        let image_ids = ids.into_iter().filter_map(|p| p.value).collect();
        let output = self
            .client
            .describe_images()
            .set_image_ids(Some(image_ids))
            .send()
            .await?;

        let images = output.images.unwrap_or_default();
    }

```



```
        if images.is_empty() {
            Err(EC2Error::new("No images for selected AMIs"))
        } else {
            Ok(images)
        }
    }
}

/// List instance types that match an image's architecture and are free tier
eligible.
pub async fn list_instance_types(&self, image: &Image) ->
Result<Vec<InstanceType>, EC2Error> {
    let architecture = format!(
        "{}",
        image.architecture().ok_or_else(|| EC2Error::new(format!(
            "Image {:?} does not have a listed architecture",
            image.image_id()
        )))?
    );
    let free_tier_eligible_filter = Filter::builder()
        .name("free-tier-eligible")
        .values("false")
        .build();
    let supported_architecture_filter = Filter::builder()
        .name("processor-info.supported-architecture")
        .values(architecture)
        .build();
    let response = self
        .client
        .describe_instance_types()
        .filters(free_tier_eligible_filter)
        .filters(supported_architecture_filter)
        .send()
        .await?;

    Ok(response
        .instance_types
        .unwrap_or_default()
        .into_iter()
        .filter_map(|iti| iti.instance_type)
        .collect())
}

pub async fn create_instance<'a>(
    &self,
```

```
        image_id: &'a str,
        instance_type: InstanceType,
        key_pair: &'a KeyPairInfo,
        security_groups: Vec<&'a SecurityGroup>,
    ) -> Result<String, EC2Error> {
        let run_instances = self
            .client
            .run_instances()
            .image_id(image_id)
            .instance_type(instance_type)
            .key_name(
                key_pair
                    .key_name()
                    .ok_or_else(|| EC2Error::new("Missing key name when launching
instance"))?),
            )
            .set_security_group_ids(Some(
                security_groups
                    .iter()
                    .filter_map(|sg| sg.group_id.clone())
                    .collect(),
            ))
            .min_count(1)
            .max_count(1)
            .send()
            .await?;

        if run_instances.instances().is_empty() {
            return Err(EC2Error::new("Failed to create instance"));
        }

        let instance_id = run_instances.instances()[0].instance_id().unwrap();
        let response = self
            .client
            .create_tags()
            .resources(instance_id)
            .tags(
                Tag::builder()
                    .key("Name")
                    .value("From SDK Examples")
                    .build(),
            )
            .send()
            .await;
```

```

    match response {
        Ok(_) => tracing::info!("Created {instance_id} and applied tags."),
        Err(err) => {
            tracing::info!("Error applying tags to {instance_id}: {err:?}");
            return Err(err.into());
        }
    }

    tracing::info!("Instance is created.");

    Ok(instance_id.to_string())
}

/// Wait for an instance to be ready and status ok (default wait 60 seconds)
pub async fn wait_for_instance_ready(
    &self,
    instance_id: &str,
    duration: Option<Duration>,
) -> Result<(), EC2Error> {
    self.client
        .wait_until_instance_status_ok()
        .instance_ids(instance_id)
        .wait(duration.unwrap_or(Duration::from_secs(60)))
        .await
        .map_err(|err| match err {
            WaiterError::ExceededMaxWait(exceeded) => EC2Error(format!(
                "Exceeded max time ({}s) waiting for instance to start.",
                exceeded.max_wait().as_secs()
            )),
            _ => EC2Error::from(err),
        })?;
    Ok(())
}

pub async fn describe_instance(&self, instance_id: &str) -> Result<Instance,
EC2Error> {
    let response = self
        .client
        .describe_instances()
        .instance_ids(instance_id)
        .send()
        .await?;

```

```
        let instance = response
            .reservations()
            .first()
            .ok_or_else(|| EC2Error::new(format!("No instance reservations for
{instance_id}")))?
            .instances()
            .first()
            .ok_or_else(|| {
                EC2Error::new(format!("No instances in reservation for
{instance_id}"))
            })?;

        Ok(instance.clone())
    }

    pub async fn start_instance(&self, instance_id: &str) -> Result<(), EC2Error>
    {
        tracing::info!("Starting instance {instance_id}");

        self.client
            .start_instances()
            .instance_ids(instance_id)
            .send()
            .await?;

        tracing::info!("Started instance.");

        Ok(())
    }

    pub async fn stop_instance(&self, instance_id: &str) -> Result<(), EC2Error>
    {
        tracing::info!("Stopping instance {instance_id}");

        self.client
            .stop_instances()
            .instance_ids(instance_id)
            .send()
            .await?;

        self.wait_for_instance_stopped(instance_id, None).await?;

        tracing::info!("Stopped instance.");
    }
}
```

```
    Ok(())
  }

  pub async fn reboot_instance(&self, instance_id: &str) -> Result<(),
EC2Error> {
    tracing::info!("Rebooting instance {instance_id}");

    self.client
      .reboot_instances()
      .instance_ids(instance_id)
      .send()
      .await?;

    Ok(())
  }

  pub async fn wait_for_instance_stopped(
    &self,
    instance_id: &str,
    duration: Option<Duration>,
  ) -> Result<(), EC2Error> {
    self.client
      .wait_until_instance_stopped()
      .instance_ids(instance_id)
      .wait(duration.unwrap_or(Duration::from_secs(60)))
      .await
      .map_err(|err| match err {
        WaiterError::ExceededMaxWait(exceeded) => EC2Error(format!(
          "Exceeded max time ({}s) waiting for instance to stop.",
          exceeded.max_wait().as_secs(),
        )),
        _ => EC2Error::from(err),
      })?;
    Ok(())
  }

  pub async fn delete_instance(&self, instance_id: &str) -> Result<(),
EC2Error> {
    tracing::info!("Deleting instance with id {instance_id}");
    self.stop_instance(instance_id).await?;
    self.client
      .terminate_instances()
      .instance_ids(instance_id)
      .send()
```

```

        .await?;
        self.wait_for_instance_terminated(instance_id).await?;
        tracing::info!("Terminated instance with id {instance_id}");
        Ok(())
    }

    async fn wait_for_instance_terminated(&self, instance_id: &str) -> Result<(),
    EC2Error> {
        self.client
            .wait_until_instance_terminated()
            .instance_ids(instance_id)
            .wait(Duration::from_secs(60))
            .await
            .map_err(|err| match err {
                WaiterError::ExceededMaxWait(exceeded) => EC2Error(format!(
                    "Exceeded max time ({}s) waiting for instance to terminate.",
                    exceeded.max_wait().as_secs(),
                )),
                _ => EC2Error::from(err),
            })?;
        Ok(())
    }

    pub async fn allocate_ip_address(&self) -> Result<AllocateAddressOutput,
    EC2Error> {
        self.client
            .allocate_address()
            .domain(DomainType::Vpc)
            .send()
            .await
            .map_err(EC2Error::from)
    }

    pub async fn deallocate_ip_address(&self, allocation_id: &str) -> Result<(),
    EC2Error> {
        self.client
            .release_address()
            .allocation_id(allocation_id)
            .send()
            .await?;
        Ok(())
    }

    pub async fn associate_ip_address(

```

```

        &self,
        allocation_id: &str,
        instance_id: &str,
    ) -> Result<AssociateAddressOutput, EC2Error> {
        let response = self
            .client
            .associate_address()
            .allocation_id(allocation_id)
            .instance_id(instance_id)
            .send()
            .await?;
        Ok(response)
    }

    pub async fn disassociate_ip_address(&self, association_id: &str) ->
Result<(), EC2Error> {
        self.client
            .disassociate_address()
            .association_id(association_id)
            .send()
            .await?;
        Ok(())
    }
}

#[derive(Debug)]
pub struct EC2Error(String);
impl EC2Error {
    pub fn new(value: impl Into<String>) -> Self {
        EC2Error(value.into())
    }

    pub fn add_message(self, message: impl Into<String>) -> Self {
        EC2Error(format!("{}", message.into(), self.0))
    }
}

impl<T: ProvideErrorMetadata> From<T> for EC2Error {
    fn from(value: T) -> Self {
        EC2Error(format!(
            "{}: {}",
            value
                .code()
                .map(String::from)

```

```

        .unwrap_or("unknown code".into()),
    value
        .message()
        .map(String::from)
        .unwrap_or("missing reason".into()),
    ))
}
}

impl std::error::Error for EC2Error {}

impl std::fmt::Display for EC2Error {
    fn fmt(&self, f: &mut std::fmt::Formatter<'_>) -> std::fmt::Result {
        write!(f, "{}", self.0)
    }
}
}

```

A SSM estrutura serve como um ponto de controle automático para testes e suas funções envolvem SSM SDK chamadas.

```

use aws_sdk_ssm::{types::Parameter, Client};
use aws_smithy_async::future::pagination_stream::TryFlatMap;

use crate::ec2::EC2Error;

#[cfg(test)]
use mockall::automock;

#[cfg(not(test))]
pub use SSMImpl as SSM;

#[cfg(test)]
pub use MockSSMImpl as SSM;

pub struct SSMImpl {
    inner: Client,
}

#[cfg_attr(test, automock)]
impl SSMImpl {
    pub fn new(inner: Client) -> Self {

```



```

    SSMImpl { inner }
  }

  pub async fn list_path(&self, path: &str) -> Result<Vec<Parameter>, EC2Error>
  {
    let maybe_params: Vec<Result<Parameter, _>> = TryFlatMap::new(
      self.inner
        .get_parameters_by_path()
        .path(path)
        .into_paginator()
        .send(),
    )
    .flat_map(|item| item.parameters.unwrap_or_default())
    .collect()
    .await;
    // Fail on the first error
    let params = maybe_params
      .into_iter()
      .collect::<<Result<Vec<Parameter>, _>>()?;
    Ok(params)
  }
}

```

O cenário usa várias estruturas no estilo “Gerenciador” para lidar com o acesso aos recursos que são criados e excluídos em todo o cenário.

```

use aws_sdk_ec2::operation::{
    allocate_address::AllocateAddressOutput,
    associate_address::AssociateAddressOutput,
};

use crate::ec2::{EC2Error, EC2};

/// ElasticIpManager tracks the lifecycle of a public IP address, including its
/// allocation from the global pool and association with a specific instance.
#[derive(Debug, Default)]
pub struct ElasticIpManager {
    elastic_ip: Option<AllocateAddressOutput>,
    association: Option<AssociateAddressOutput>,
}

```

```
impl ElasticIpManager {
    pub fn has_allocation(&self) -> bool {
        self.elastic_ip.is_some()
    }

    pub fn public_ip(&self) -> &str {
        if let Some(allocation) = &self.elastic_ip {
            if let Some(addr) = allocation.public_ip() {
                return addr;
            }
        }
        "0.0.0.0"
    }

    pub async fn allocate(&mut self, ec2: &EC2) -> Result<(), EC2Error> {
        let allocation = ec2.allocate_ip_address().await?;
        self.elastic_ip = Some(allocation);
        Ok(())
    }

    pub async fn associate(&mut self, ec2: &EC2, instance_id: &str) -> Result<(),
    EC2Error> {
        if let Some(allocation) = &self.elastic_ip {
            if let Some(allocation_id) = allocation.allocation_id() {
                let association = ec2.associate_ip_address(allocation_id,
                instance_id).await?;
                self.association = Some(association);
                return Ok(());
            }
        }
        Err(EC2Error::new("No ip address allocation to associate"))
    }

    pub async fn remove(mut self, ec2: &EC2) -> Result<(), EC2Error> {
        if let Some(association) = &self.association {
            if let Some(association_id) = association.association_id() {
                ec2.disassociate_ip_address(association_id).await?;
            }
        }
        self.association = None;
        if let Some(allocation) = &self.elastic_ip {
            if let Some(allocation_id) = allocation.allocation_id() {
                ec2.deallocate_ip_address(allocation_id).await?;
            }
        }
    }
}
```

```
    }
    self.elastic_ip = None;
    Ok(())
  }
}

use std::fmt::Display;

use aws_sdk_ec2::types::{Instance, InstanceType, KeyPairInfo, SecurityGroup};

use crate::ec2::{EC2Error, EC2};

/// InstanceManager wraps the lifecycle of an EC2 Instance.
#[derive(Debug, Default)]
pub struct InstanceManager {
    instance: Option<Instance>,
}

impl InstanceManager {
    pub fn instance_id(&self) -> &str {
        if let Some(instance) = &self.instance {
            if let Some(id) = instance.instance_id() {
                return id;
            }
        }
        "Unknown"
    }

    pub fn instance_name(&self) -> &str {
        if let Some(instance) = &self.instance {
            if let Some(tag) = instance.tags().iter().find(|e| e.key() ==
Some("Name")) {
                if let Some(value) = tag.value() {
                    return value;
                }
            }
        }
        "Unknown"
    }

    pub fn instance_ip(&self) -> &str {
        if let Some(instance) = &self.instance {
            if let Some(public_ip_address) = instance.public_ip_address() {
```

```
        return public_ip_address;
    }
}
"0.0.0.0"
}

pub fn instance_display_name(&self) -> String {
    format!("{}", self.instance_name(), self.instance_id())
}

/// Create an EC2 instance with the given ID on a given type, using a
/// generated KeyPair and applying a list of security groups.
pub async fn create(
    &mut self,
    ec2: &EC2,
    image_id: &str,
    instance_type: InstanceType,
    key_pair: &KeyPairInfo,
    security_groups: Vec<&SecurityGroup>,
) -> Result<(), EC2Error> {
    let instance_id = ec2
        .create_instance(image_id, instance_type, key_pair, security_groups)
        .await?;
    let instance = ec2.describe_instance(&instance_id).await?;
    self.instance = Some(instance);
    Ok(())
}

/// Start the managed EC2 instance, if present.
pub async fn start(&self, ec2: &EC2) -> Result<(), EC2Error> {
    if self.instance.is_some() {
        ec2.start_instance(self.instance_id()).await?;
    }
    Ok(())
}

/// Stop the managed EC2 instance, if present.
pub async fn stop(&self, ec2: &EC2) -> Result<(), EC2Error> {
    if self.instance.is_some() {
        ec2.stop_instance(self.instance_id()).await?;
    }
    Ok(())
}
```

```

pub async fn reboot(&self, ec2: &EC2) -> Result<(), EC2Error> {
    if self.instance.is_some() {
        ec2.reboot_instance(self.instance_id()).await?;
        ec2.wait_for_instance_stopped(self.instance_id(), None)
            .await?;
        ec2.wait_for_instance_ready(self.instance_id(), None)
            .await?;
    }
    Ok(())
}

/// Terminate and delete the managed EC2 instance, if present.
pub async fn delete(self, ec2: &EC2) -> Result<(), EC2Error> {
    if self.instance.is_some() {
        ec2.delete_instance(self.instance_id()).await?;
    }
    Ok(())
}
}

impl Display for InstanceManager {
    fn fmt(&self, f: &mut std::fmt::Formatter<'_>) -> std::fmt::Result {
        if let Some(instance) = &self.instance {
            writeln!(f, "\tID: {}",
instance.instance_id().unwrap_or("(Unknown)"));
            writeln!(
                f,
                "\tImage ID: {}",
                instance.image_id().unwrap_or("(Unknown)")
            );
            writeln!(
                f,
                "\tInstance type: {}",
                instance
                    .instance_type()
                    .map(|it| format!("{it}"))
                    .unwrap_or("(Unknown)".to_string())
            );
            writeln!(
                f,
                "\tKey name: {}",
                instance.key_name().unwrap_or("(Unknown)")
            );
        }
    }
}

```

```

        writeln!(f, "\tVPC ID: {}",
instance.vpc_id().unwrap_or("(Unknown)"));
        writeln!(
            f,
            "\tPublic IP: {}",
            instance.public_ip_address().unwrap_or("(Unknown)")
        );
        let instance_state = instance
            .state
            .as_ref()
            .map(|is| {
                is.name()
                    .map(|isn| format!("{isn}"))
                    .unwrap_or("(Unknown)".to_string())
            })
            .unwrap_or("(Unknown)".to_string());
        writeln!(f, "\tState: {instance_state}");
    } else {
        writeln!(f, "\tNo loaded instance");
    }
    Ok(())
}
}

use std::{env, path::PathBuf};

use aws_sdk_ec2::types::KeyPairInfo;

use crate::ec2::{EC2Error, EC2};

use super::util::Util;

/// KeyPairManager tracks a KeyPairInfo and the path the private key has been
/// written to, if it's been created.
#[derive(Debug)]
pub struct KeyPairManager {
    key_pair: KeyPairInfo,
    key_file_path: Option<PathBuf>,
    key_file_dir: PathBuf,
}

impl KeyPairManager {
    pub fn new() -> Self {

```

```
        Self::default()
    }

    pub fn key_pair(&self) -> &KeyPairInfo {
        &self.key_pair
    }

    pub fn key_file_path(&self) -> Option<&PathBuf> {
        self.key_file_path.as_ref()
    }

    pub fn key_file_dir(&self) -> &PathBuf {
        &self.key_file_dir
    }

    /// 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.
    pub async fn create(
        &mut self,
        ec2: &EC2,
        util: &Util,
        key_name: String,
    ) -> Result<KeyPairInfo, EC2Error> {
        let (key_pair, material) = ec2
            .create_key_pair(key_name.clone())
            .await
            .map_err(|e| e.add_message(format!("Couldn't create key
{key_name}")))?;

        let path = self.key_file_dir.join(format!("{key_name}.pem"));

        util.write_secure(&key_name, &path, material)?;

        self.key_file_path = Some(path);
        self.key_pair = key_pair.clone();

        Ok(key_pair)
    }
}
```

```

    pub async fn delete(self, ec2: &EC2, util: &Util) -> Result<(), EC2Error> {
        if let Some(key_pair_id) = self.key_pair.key_pair_id() {
            ec2.delete_key_pair(key_pair_id).await?;
            if let Some(key_path) = self.key_file_path() {
                if let Err(err) = util.remove(key_path) {
                    eprintln!("Failed to remove {key_path:?} ({err:?})");
                }
            }
        }
        Ok(())
    }

    pub async fn list(&self, ec2: &EC2) -> Result<Vec<KeyPairInfo>, EC2Error> {
        ec2.list_key_pair().await
    }
}

impl Default for KeyPairManager {
    fn default() -> Self {
        KeyPairManager {
            key_pair: KeyPairInfo::builder().build(),
            key_file_path: Default::default(),
            key_file_dir: env::temp_dir(),
        }
    }
}

use std::net::Ipv4Addr;

use aws_sdk_ec2::types::SecurityGroup;

use crate::ec2::{EC2Error, EC2};

/// SecurityGroupManager tracks the lifecycle of a SecurityGroup for an instance,
/// including adding a rule to allow SSH from a public IP address.
#[derive(Debug, Default)]
pub struct SecurityGroupManager {
    group_name: String,
    group_description: String,
    security_group: Option<SecurityGroup>,
}

impl SecurityGroupManager {

```



```
pub async fn create(
    &mut self,
    ec2: &EC2,
    group_name: &str,
    group_description: &str,
) -> Result<(), EC2Error> {
    self.group_name = group_name.into();
    self.group_description = group_description.into();

    self.security_group = Some(
        ec2.create_security_group(group_name, group_description)
            .await
            .map_err(|e| e.add_message("Couldn't create security group"))?,
    );

    Ok(())
}

pub async fn authorize_ingress(&self, ec2: &EC2, ip_address: Ipv4Addr) ->
Result<(), EC2Error> {
    if let Some(sg) = &self.security_group {
        ec2.authorize_security_group_ssh_ingress(
            sg.group_id()
                .ok_or_else(|| EC2Error::new("Missing security group ID"))?,
            vec![ip_address],
        )
        .await?;
    }

    Ok(())
}

pub async fn delete(self, ec2: &EC2) -> Result<(), EC2Error> {
    if let Some(sg) = &self.security_group {
        ec2.delete_security_group(
            sg.group_id()
                .ok_or_else(|| EC2Error::new("Missing security group ID"))?,
        )
        .await?;
    }

    Ok(())
}
```

```

pub fn group_name(&self) -> &str {
    &self.group_name
}

pub fn vpc_id(&self) -> Option<&str> {
    self.security_group.as_ref().and_then(|sg| sg.vpc_id())
}

pub fn security_group(&self) -> Option<&SecurityGroup> {
    self.security_group.as_ref()
}
}

impl std::fmt::Display for SecurityGroupManager {
    fn fmt(&self, f: &mut std::fmt::Formatter<'_>) -> std::fmt::Result {
        match &self.security_group {
            Some(sg) => {
                writeln!(
                    f,
                    "Security group: {}",
                    sg.group_name().unwrap_or("unknown group")
                )?;
                writeln!(f, "\tID: {}", sg.group_id().unwrap_or("unknown group
id")))?;
                writeln!(f, "\tVPC: {}", sg.vpc_id().unwrap_or("unknown group
vpc")))?;
                if !sg.ip_permissions().is_empty() {
                    writeln!(f, "\tInbound Permissions:");
                    for permission in sg.ip_permissions() {
                        writeln!(f, "\t\t{permission:?}")?;
                    }
                }
                Ok(())
            }
            None => writeln!(f, "No security group loaded."),
        }
    }
}
}

```

O principal ponto de entrada para o cenário.

```
use ec2_code_examples::{
    ec2::EC2,
    getting_started::{
        scenario::{run, Ec2InstanceScenario},
        util::UtilImpl,
    },
    ssm::SSM,
};

#[tokio::main]
async fn main() {
    tracing_subscriber::fmt::init();
    let sdk_config = aws_config::load_from_env().await;
    let ec2 = EC2::new(aws_sdk_ec2::Client::new(&sdk_config));
    let ssm = SSM::new(aws_sdk_ssm::Client::new(&sdk_config));
    let util = UtilImpl {};
    let scenario = Ec2InstanceScenario::new(ec2, ssm, util);
    run(scenario).await;
}
```

- Para API obter detalhes, consulte os tópicos a seguir AWS SDK para API referência do Rust.

- [AllocateAddress](#)
- [AssociateAddress](#)
- [AuthorizeSecurityGroupIngress](#)
- [CreateKeyPair](#)
- [CreateSecurityGroup](#)
- [DeleteKeyPair](#)
- [DeleteSecurityGroup](#)
- [DescribeImages](#)
- [DescribeInstanceTypes](#)
- [DescribeInstances](#)
- [DescribeKeyPairs](#)
- [DescribeSecurityGroups](#)
- [DisassociateAddress](#)
- [ReleaseAddress](#)

- [StartInstances](#)
- [StopInstances](#)
- [TerminateInstances](#)
- [UnmonitorInstances](#)

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Ações para a Amazon EC2 usando AWS SDKs

Os exemplos de código a seguir demonstram como realizar EC2 ações individuais da Amazon com AWS SDKs. Cada exemplo inclui um link para GitHub, onde você pode encontrar instruções para configurar e executar o código.

Esses trechos chamam Amazon EC2 API e são trechos de código de programas maiores que devem ser executados em contexto. Você pode ver as ações em contexto em [Cenários para o EC2 uso da Amazon AWS SDKs](#).

Os exemplos a seguir incluem apenas as ações mais utilizadas. Para obter uma lista completa, consulte a [Amazon Elastic Compute Cloud API Reference](#).

Exemplos

- [Use AcceptVpcPeeringConnection com um CLI](#)
- [Use AllocateAddress com um AWS SDK ou CLI](#)
- [Use AllocateHosts com um CLI](#)
- [Use AssignPrivateIpAddresses com um CLI](#)
- [Use AssociateAddress com um AWS SDK ou CLI](#)
- [Use AssociateDhcpOptions com um CLI](#)
- [Use AssociateRouteTable com um CLI](#)
- [Use AttachInternetGateway com um CLI](#)
- [Use AttachNetworkInterface com um CLI](#)
- [Use AttachVolume com um CLI](#)
- [Use AttachVpnGateway com um CLI](#)

- [Use AuthorizeSecurityGroupEgress com um CLI](#)
- [Use AuthorizeSecurityGroupIngress com um AWS SDK ou CLI](#)
- [Use CancelCapacityReservation com um CLI](#)
- [Use CancellImportTask com um CLI](#)
- [Use CancelSpotFleetRequests com um CLI](#)
- [Use CancelSpotInstanceRequests com um CLI](#)
- [Use ConfirmProductInstance com um CLI](#)
- [Use CopyImage com um CLI](#)
- [Use CopySnapshot com um CLI](#)
- [Use CreateCapacityReservation com um CLI](#)
- [Use CreateCustomerGateway com um CLI](#)
- [Use CreateDhcpOptions com um CLI](#)
- [Use CreateFlowLogs com um CLI](#)
- [Use CreateImage com um CLI](#)
- [Use CreateInstanceExportTask com um CLI](#)
- [Use CreateInternetGateway com um CLI](#)
- [Use CreateKeyPair com um AWS SDK ou CLI](#)
- [Use CreateLaunchTemplate com um AWS SDK ou CLI](#)
- [Use CreateNetworkAcl com um CLI](#)
- [Use CreateNetworkAclEntry com um CLI](#)
- [Use CreateNetworkInterface com um CLI](#)
- [Use CreatePlacementGroup com um CLI](#)
- [Use CreateRoute com um CLI](#)
- [Use CreateRouteTable com um AWS SDK ou CLI](#)
- [Use CreateSecurityGroup com um AWS SDK ou CLI](#)
- [Use CreateSnapshot com um CLI](#)
- [Use CreateSpotDatafeedSubscription com um CLI](#)
- [Use CreateSubnet com um AWS SDK ou CLI](#)

- [Use CreateTags com um AWS SDK ou CLI](#)
- [Use CreateVolume com um CLI](#)
- [Use CreateVpc com um AWS SDK ou CLI](#)
- [Use CreateVpcEndpoint com um AWS SDK ou CLI](#)
- [Use CreateVpnConnection com um CLI](#)
- [Use CreateVpnConnectionRoute com um CLI](#)
- [Use CreateVpnGateway com um CLI](#)
- [Use DeleteCustomerGateway com um CLI](#)
- [Use DeleteDhcpOptions com um CLI](#)
- [Use DeleteFlowLogs com um CLI](#)
- [Use DeleteInternetGateway com um CLI](#)
- [Use DeleteKeyPair com um AWS SDK ou CLI](#)
- [Use DeleteLaunchTemplate com um AWS SDK ou CLI](#)
- [Use DeleteNetworkAcl com um CLI](#)
- [Use DeleteNetworkAclEntry com um CLI](#)
- [Use DeleteNetworkInterface com um CLI](#)
- [Use DeletePlacementGroup com um CLI](#)
- [Use DeleteRoute com um CLI](#)
- [Use DeleteRouteTable com um CLI](#)
- [Use DeleteSecurityGroup com um AWS SDK ou CLI](#)
- [Use DeleteSnapshot com um AWS SDK ou CLI](#)
- [Use DeleteSpotDatafeedSubscription com um CLI](#)
- [Use DeleteSubnet com um CLI](#)
- [Use DeleteTags com um CLI](#)
- [Use DeleteVolume com um CLI](#)
- [Use DeleteVpc com um AWS SDK ou CLI](#)
- [Use DeleteVpcEndpoint com um AWS SDK](#)
- [Use DeleteVpnConnection com um CLI](#)

- [Use DeleteVpnConnectionRoute com um CLI](#)
- [Use DeleteVpnGateway com um CLI](#)
- [Use DeregisterImage com um CLI](#)
- [Use DescribeAccountAttributes com um CLI](#)
- [Use DescribeAddresses com um AWS SDK ou CLI](#)
- [Use DescribeAvailabilityZones com um AWS SDK ou CLI](#)
- [Use DescribeBundleTasks com um CLI](#)
- [Use DescribeCapacityReservations com um CLI](#)
- [Use DescribeCustomerGateways com um CLI](#)
- [Use DescribeDhcpOptions com um CLI](#)
- [Use DescribeFlowLogs com um CLI](#)
- [Use DescribeHostReservationOfferings com um CLI](#)
- [Use DescribeHosts com um CLI](#)
- [Use DescribelamInstanceProfileAssociations com um AWS SDK ou CLI](#)
- [Use DescribelIdFormat com um CLI](#)
- [Use DescribelIdentityIdFormat com um CLI](#)
- [Use DescribelImageAttribute com um CLI](#)
- [Use DescribelImages com um AWS SDK ou CLI](#)
- [Use DescribelImportImageTasks com um CLI](#)
- [Use DescribelImportSnapshotTasks com um CLI](#)
- [Use DescribelInstanceAttribute com um CLI](#)
- [Use DescribelInstanceStatus com um AWS SDK ou CLI](#)
- [Use DescribelInstanceTypes com um AWS SDK ou CLI](#)
- [Use DescribelInstances com um AWS SDK ou CLI](#)
- [Use DescribelInternetGateways com um CLI](#)
- [Use DescribeKeyPairs com um AWS SDK ou CLI](#)
- [Use DescribeNetworkAcls com um CLI](#)
- [Use DescribeNetworkInterfaceAttribute com um CLI](#)

- [Use DescribeNetworkInterfaces com um CLI](#)
- [Use DescribePlacementGroups com um CLI](#)
- [Use DescribePrefixLists com um CLI](#)
- [Use DescribeRegions com um AWS SDK ou CLI](#)
- [Use DescribeRouteTables com um AWS SDK ou CLI](#)
- [Use DescribeScheduledInstanceAvailability com um CLI](#)
- [Use DescribeScheduledInstances com um CLI](#)
- [Use DescribeSecurityGroups com um AWS SDK ou CLI](#)
- [Use DescribeSnapshotAttribute com um CLI](#)
- [Use DescribeSnapshots com um AWS SDK ou CLI](#)
- [Use DescribeSpotDatafeedSubscription com um CLI](#)
- [Use DescribeSpotFleetInstances com um CLI](#)
- [Use DescribeSpotFleetRequestHistory com um CLI](#)
- [Use DescribeSpotFleetRequests com um CLI](#)
- [Use DescribeSpotInstanceRequests com um CLI](#)
- [Use DescribeSpotPriceHistory com um CLI](#)
- [Use DescribeSubnets com um AWS SDK ou CLI](#)
- [Use DescribeTags com um CLI](#)
- [Use DescribeVolumeAttribute com um CLI](#)
- [Use DescribeVolumeStatus com um CLI](#)
- [Use DescribeVolumes com um CLI](#)
- [Use DescribeVpcAttribute com um CLI](#)
- [Use DescribeVpcClassicLink com um CLI](#)
- [Use DescribeVpcClassicLinkDnsSupport com um CLI](#)
- [Use DescribeVpcEndpointServices com um CLI](#)
- [Use DescribeVpcEndpoints com um CLI](#)
- [Use DescribeVpcs com um AWS SDK ou CLI](#)
- [Use DescribeVpnConnections com um CLI](#)
- [Use DescribeVpnGateways com um CLI](#)

- [Use DetachInternetGateway com um CLI](#)
- [Use DetachNetworkInterface com um CLI](#)
- [Use DetachVolume com um CLI](#)
- [Use DetachVpnGateway com um CLI](#)
- [Use DisableVgwRoutePropagation com um CLI](#)
- [Use DisableVpcClassicLink com um CLI](#)
- [Use DisableVpcClassicLinkDnsSupport com um CLI](#)
- [Use DisassociateAddress com um AWS SDK ou CLI](#)
- [Use DisassociateRouteTable com um CLI](#)
- [Use EnableVgwRoutePropagation com um CLI](#)
- [Use EnableVolumelo com um CLI](#)
- [Use EnableVpcClassicLink com um CLI](#)
- [Use EnableVpcClassicLinkDnsSupport com um CLI](#)
- [Use GetConsoleOutput com um CLI](#)
- [Use GetHostReservationPurchasePreview com um CLI](#)
- [Use GetPasswordData com um AWS SDK ou CLI](#)
- [Use ImportImage com um CLI](#)
- [Use ImportKeyPair com um CLI](#)
- [Use ImportSnapshot com um CLI](#)
- [Use ModifyCapacityReservation com um CLI](#)
- [Use ModifyHosts com um CLI](#)
- [Use ModifyIdFormat com um CLI](#)
- [Use ModifyImageAttribute com um CLI](#)
- [Use ModifyInstanceAttribute com um CLI](#)
- [Use ModifyInstanceCreditSpecification com um CLI](#)
- [Use ModifyNetworkInterfaceAttribute com um CLI](#)
- [Use ModifyReservedInstances com um CLI](#)
- [Use ModifySnapshotAttribute com um CLI](#)
- [Use ModifySpotFleetRequest com um CLI](#)

- [Use ModifySubnetAttribute com um CLI](#)
- [Use ModifyVolumeAttribute com um CLI](#)
- [Use ModifyVpcAttribute com um CLI](#)
- [Use MonitorInstances com um AWS SDK ou CLI](#)
- [Use MoveAddressToVpc com um CLI](#)
- [Use PurchaseHostReservation com um CLI](#)
- [Use PurchaseScheduledInstances com um CLI](#)
- [Use RebootInstances com um AWS SDK ou CLI](#)
- [Use RegisterImage com um CLI](#)
- [Use RejectVpcPeeringConnection com um CLI](#)
- [Use ReleaseAddress com um AWS SDK ou CLI](#)
- [Use ReleaseHosts com um CLI](#)
- [Use ReplacelamInstanceProfileAssociation com um AWS SDK ou CLI](#)
- [Use ReplaceNetworkAclAssociation com um CLI](#)
- [Use ReplaceNetworkAclEntry com um CLI](#)
- [Use ReplaceRoute com um CLI](#)
- [Use ReplaceRouteTableAssociation com um CLI](#)
- [Use ReportInstanceStatus com um CLI](#)
- [Use RequestSpotFleet com um CLI](#)
- [Use RequestSpotInstances com um CLI](#)
- [Use ResetImageAttribute com um CLI](#)
- [Use ResetInstanceAttribute com um CLI](#)
- [Use ResetNetworkInterfaceAttribute com um CLI](#)
- [Use ResetSnapshotAttribute com um CLI](#)
- [Use RevokeSecurityGroupEgress com um CLI](#)
- [Use RevokeSecurityGroupIngress com um CLI](#)
- [Use RunInstances com um AWS SDK ou CLI](#)
- [Use RunScheduledInstances com um CLI](#)
- [Use StartInstances com um AWS SDK ou CLI](#)

- [Use StopInstances com um AWS SDK ou CLI](#)
- [Use TerminateInstances com um AWS SDK ou CLI](#)
- [Use UnassignPrivateIpAddresses com um CLI](#)
- [Use UnmonitorInstances com um AWS SDK ou CLI](#)
- [Use UpdateSecurityGroupRuleDescriptionsIngress com um CLI](#)

Use `AcceptVpcPeeringConnection` com um CLI

Os exemplos de código a seguir mostram como usar o `AcceptVpcPeeringConnection`.

CLI

AWS CLI

Para aceitar uma conexão de VPC emparelhamento

Este exemplo aceita a solicitação de conexão de VPC emparelhamento especificada.

Comando:

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

Saída:

```
{
  "VpcPeeringConnection": {
    "Status": {
      "Message": "Provisioning",
      "Code": "provisioning"
    },
    "Tags": [],
    "AcceptorVpcInfo": {
      "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"
  }
}
}

```

- Para API obter detalhes, consulte [AcceptVpcPeeringConnection](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo aprova o `pcx-1dfad234b56ff78be` solicitado `VpcPeeringConnectionId`

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

Saída:

```

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

```

- Para API obter detalhes, consulte [AcceptVpcPeeringConnection](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **AllocateAddress** com um AWS SDK ou CLI

Os exemplos de código a seguir mostram como usar o `AllocateAddress`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação no contexto no seguinte exemplo de código:

- [Aprender os conceitos básicos](#)

.NET

AWS SDK for .NET

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
/// <summary>
/// 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.
/// </summary>
/// <returns>The response object for the allocated address.</returns>
public async Task<AllocateAddressResponse> AllocateAddress()
{
    var request = new AllocateAddressRequest();

    try
    {
        var response = await _amazonEC2.AllocateAddressAsync(request);
        Console.WriteLine($"Allocated IP: {response.PublicIp} with allocation
ID {response.AllocationId}.");
        return response;
    }
    catch (AmazonEC2Exception ec2Exception)
    {
        if (ec2Exception.ErrorCode == "AddressLimitExceeded")
        {
            // For more information on Elastic IP address quotas, see:
            // https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/elastic-
ip-addresses-eip.html#using-instance-addressing-limit
            _logger.LogError($"Unable to allocate Elastic IP, address limit
exceeded. {ec2Exception.Message}");
        }

        throw;
    }
}
```

```

    }
    catch (Exception ex)
    {
        _logger.LogError($"An error occurred while allocating Elastic IP.:
{ex.Message}");
        throw;
    }
}

```

- Para API obter detalhes, consulte [AllocateAddress](#) em AWS SDK for .NET API Referência.

Bash

AWS CLI com script Bash

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```

#####
# 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
}

```

As funções utilitárias usadas neste exemplo.

```

#####
# 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
}

```

- Para API obter detalhes, consulte [AllocateAddress](#) na Referência de AWS CLI Comandos.

C++

SDK para C++

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```

//! Allocate an Elastic IP address and associate it with an Amazon Elastic
  Compute Cloud
//! (Amazon EC2) instance.
/*!
  \param instanceID: An EC2 instance ID.
  \param[out] publicIPAddress: String to return the public IP address.
  \param[out] allocationID: String to return the allocation ID.
  \param clientConfiguration: AWS client configuration.
  \return bool: Function succeeded.
*/
bool AwsDoc::EC2::allocateAndAssociateAddress(const Aws::String &instanceId,
  Aws::String &publicIPAddress,
  Aws::String &allocationID,
  const
  Aws::Client::ClientConfiguration &clientConfiguration) {

```

```
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;
}
const Aws::EC2::Model::AllocateAddressResponse &response =
outcome.GetResult();
allocationID = response.GetAllocationId();
publicIPAddress = response.GetPublicIp();

return true;
}
```

- Para API obter detalhes, consulte [AllocateAddress](#) em AWS SDK for C++ API Referência.

CLI

AWS CLI

Exemplo 1: alocar um endereço IP elástico do conjunto de endereços da Amazon

O exemplo `allocate-address` a seguir aloca um endereço IP elástico. A Amazon EC2 seleciona o endereço do pool de endereços da Amazon.

```
aws ec2 allocate-address
```

Saída:

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

```
"Domain": "vpc"
}
```

Para obter mais informações, consulte [Endereços IP elásticos](#) no Guia EC2 do usuário da Amazon.

Exemplo 2: alocar um endereço IP elástico e associá-lo a um grupo de borda de rede

O exemplo `allocate-address` a seguir aloca um endereço IP elástico e o associa ao grupo de borda de rede especificado.

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

Saída:

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

Para obter mais informações, consulte [Endereços IP elásticos](#) no Guia EC2 do usuário da Amazon.

Exemplo 3: alocar um endereço IP elástico de um conjunto de endereços de sua propriedade

O exemplo `allocate-address` a seguir aloca um endereço IP elástico de um conjunto de endereços que você trouxe para a conta da Amazon Web Services. A Amazon EC2 seleciona o endereço do pool de endereços.

```
aws ec2 allocate-address \
  --public-ipv4-pool ipv4pool-ec2-1234567890abcdef0
```

Saída:

```
{
  "AllocationId": "eipalloc-02463d08ceEXAMPLE",
  "NetworkBorderGroup": "us-west-2",
}
```

```
"CustomerOwnedIp": "18.218.95.81",
"CustomerOwnedIpv4Pool": "ipv4pool-ec2-1234567890abcdef0",
"Domain": "vpc"
"NetworkBorderGroup": "us-west-2",
}
```

Para obter mais informações, consulte [Endereços IP elásticos](#) no Guia EC2 do usuário da Amazon.

- Para API obter detalhes, consulte [AllocateAddress](#) na Referência de AWS CLI Comandos.

Java

SDK para Java 2.x

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
/**
 * Allocates an Elastic IP address asynchronously in the VPC domain.
 *
 * @return a {@link CompletableFuture} containing the allocation ID of the
 * allocated Elastic IP address
 */
public CompletableFuture<String> allocateAddressAsync() {
    AllocateAddressRequest allocateRequest = AllocateAddressRequest.builder()
        .domain(DomainType.VPC)
        .build();

    CompletableFuture<AllocateAddressResponse> responseFuture =
getAsyncClient().allocateAddress(allocateRequest);
    return
responseFuture.thenApply(AllocateAddressResponse::allocationId).whenComplete((result,
ex) -> {
        if (ex != null) {
            throw new RuntimeException("Failed to allocate address", ex);
        }
    });
}
```

- Para API obter detalhes, consulte [AllocateAddress](#) em AWS SDK for Java 2.x API Referência.

JavaScript

SDK para JavaScript (v3)

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
import { AllocateAddressCommand, EC2Client } from "@aws-sdk/client-ec2";

/**
 * Allocates an Elastic IP address to your AWS account.
 */
export const main = async () => {
  const client = new EC2Client({});
  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: ${PublicIp}`);
    console.log(
      "You can view your IP addresses in the AWS Management Console for Amazon EC2. Look under Network & Security > Elastic IPs",
    );
  } catch (caught) {
    if (caught instanceof Error && caught.name === "MissingParameter") {
      console.warn(`${caught.message}. Did you provide these values?`);
    } else {
      throw caught;
    }
  }
};
```

```
import { fileURLToPath } from "node:url";
// Call function if run directly.
if (process.argv[1] === fileURLToPath(import.meta.url)) {
  main();
}
```

- Para API obter detalhes, consulte [AllocateAddress](#) em AWS SDK for JavaScript API Referência.

Kotlin

SDK para Kotlin

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
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
    }
}
```

- Para API obter detalhes, consulte a [AllocateAddress](#) referência AWS SDKdo Kotlin API.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo aloca um endereço IP elástico para usar com uma instância em a VPC

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

Saída:

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

Exemplo 2: Este exemplo aloca um endereço IP elástico para usar com uma instância em EC2 -Classic.

```
New-EC2Address
```

Saída:

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

- Para API obter detalhes, consulte [AllocateAddress](#) em Referência de AWS Tools for PowerShell cmdlet.

Python

SDKpara Python (Boto3)

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```

class ElasticIpWrapper:
    """Encapsulates Amazon Elastic Compute Cloud (Amazon EC2) Elastic IP address
    actions using the client interface."""

    class ElasticIp:
        """Represents an Elastic IP and its associated instance."""

        def __init__(
            self, allocation_id: str, public_ip: str, instance_id: Optional[str]
            = None
        ) -> None:
            """
            Initializes the ElasticIp object.

            :param allocation_id: The allocation ID of the Elastic IP.
            :param public_ip: The public IP address of the Elastic IP.
            :param instance_id: The ID of the associated EC2 instance, if any.
            """
            self.allocation_id = allocation_id
            self.public_ip = public_ip
            self.instance_id = instance_id

        def __init__(self, ec2_client: Any) -> None:
            """
            Initializes the ElasticIpWrapper with an EC2 client.

            :param ec2_client: A Boto3 Amazon EC2 client. This client provides low-
            level
                access to AWS EC2 services.
            """
            self.ec2_client = ec2_client
            self.elastic_ips: List[ElasticIpWrapper.ElasticIp] = []

    @classmethod
    def from_client(cls) -> "ElasticIpWrapper":
        """
        Creates an ElasticIpWrapper instance with a default EC2 client.

        :return: An instance of ElasticIpWrapper initialized with the default EC2
        client.
        """
        ec2_client = boto3.client("ec2")
        return cls(ec2_client)

```



```
def allocate(self) -> "ElasticIpWrapper.ElasticIp":
    """
    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 ElasticIp object for the newly created Elastic IP address.
    :raises ClientError: If the allocation fails, such as reaching the
    maximum limit of Elastic IPs.
    """
    try:
        response = self.ec2_client.allocate_address(Domain="vpc")
        elastic_ip = self.ElasticIp(
            allocation_id=response["AllocationId"],
            public_ip=response["PublicIp"]
        )
        self.elastic_ips.append(elastic_ip)
    except ClientError as err:
        if err.response["Error"]["Code"] == "AddressLimitExceeded":
            logger.error(
                "Max IP's reached. Release unused addresses or contact AWS
                Support for an increase."
            )
            raise err
    return elastic_ip
```

- Para API obter detalhes, consulte a [AllocateAddress](#) Referência AWS SDK do Python (Boto3). API

Ruby

SDK para Ruby

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
# 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')
  response.allocation_id
rescue StandardError => e
  puts "Error allocating Elastic IP address: #{e.message}"
  'Error'
end
```

- Para API obter detalhes, consulte [AllocateAddress](#) em AWS SDK for Ruby API Referência.

Rust

SDK para Rust

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
pub async fn allocate_ip_address(&self) -> Result<AllocateAddressOutput,
EC2Error> {
    self.client
```

```

        .allocate_address()
        .domain(DomainType::Vpc)
        .send()
        .await
        .map_err(EC2Error::from)
    }

```

- Para API obter detalhes, consulte [AllocateAddress](#) AWS SDKAPIreferência do Rust.

SAP ABAP

SDKpara SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```

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.

```

- Para API obter detalhes, consulte [AllocateAddress AWSSDKpara SAP ABAP API](#) referência.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use `AllocateHosts` com um CLI

Os exemplos de código a seguir mostram como usar o `AllocateHosts`.

CLI

AWS CLI

Exemplo 1: Para alocar um host dedicado

O `allocate-hosts` exemplo a seguir aloca um único host dedicado na zona de `eu-west-1a` disponibilidade, no qual você pode executar `m5.large` instâncias. Por padrão, o host dedicado aceita somente a execução da instância de destino e não oferece suporte à recuperação do host.

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

Saída:

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

Exemplo 2: Para alocar um host dedicado com posicionamento automático e recuperação de host ativados

O `allocate-hosts` exemplo a seguir aloca um único host dedicado na zona de `eu-west-1a` disponibilidade com posicionamento automático e recuperação de host ativados.

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

Saída:

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

Exemplo 3: Para alocar um host dedicado com tags

O `allocate-hosts` exemplo a seguir aloca um único host dedicado e aplica uma tag com uma chave chamada `purpose` e um valor de `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}'
```

Saída:

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

Para obter mais informações, consulte [Alocação de hosts dedicados](#) no Guia do usuário do Amazon Elastic Compute Cloud para instâncias Linux.

- Para API obter detalhes, consulte [AllocateHosts](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: este exemplo aloca um host dedicado à sua conta para o tipo de instância e zona de disponibilidade específicos

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

Saída:

```
h-01e23f4cd567890f3
```

- Para API obter detalhes, consulte [AllocateHosts](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use `AssignPrivateIpAddresses` com um CLI

Os exemplos de código a seguir mostram como usar o `AssignPrivateIpAddresses`.

CLI

AWS CLI

Para atribuir um endereço IP privado secundário específico a uma interface de rede

Este exemplo atribui o endereço IP privado secundário especificado à interface de rede especificada. Se o comando for bem-sucedido, nenhuma saída será retornada.

Comando:

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

Para atribuir endereços IP privados secundários que a Amazon EC2 seleciona a uma interface de rede

Este exemplo atribui dois endereços IP privados secundários à interface de rede especificada. A Amazon atribui EC2 automaticamente esses endereços IP a partir dos endereços IP disponíveis no intervalo de CIDR blocos da sub-rede à qual a interface de rede está associada. Se o comando for bem-sucedido, nenhuma saída será retornada.

Comando:

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

- Para API obter detalhes, consulte [AssignPrivateIpAddresses](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo atribui o endereço IP privado secundário especificado à interface de rede especificada.

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

Exemplo 2: Este exemplo cria dois endereços IP privados secundários e os atribui à interface de rede especificada.

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

- Para API obter detalhes, consulte [AssignPrivateIpAddresses](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **AssociateAddress** com um AWS SDK ou CLI

Os exemplos de código a seguir mostram como usar o AssociateAddress.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação no contexto no seguinte exemplo de código:

- [Aprender os conceitos básicos](#)

.NET

AWS SDK for .NET

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
/// <summary>
/// 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.
/// </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)
{
    try
    {
        var request = new AssociateAddressRequest
        {
            AllocationId = allocationId,
            InstanceId = instanceId
        };

        var response = await _amazonEC2.AssociateAddressAsync(request);
        return response.AssociationId;
    }
    catch (AmazonEC2Exception ec2Exception)
    {
        if (ec2Exception.ErrorCode == "InvalidInstanceId")
        {
            _logger.LogError(
```



```

        $"InstanceId is invalid, unable to associate address.
{ec2Exception.Message}");
    }

    throw;
}
catch (Exception ex)
{
    _logger.LogError(
        $"An error occurred while associating the Elastic IP.:
{ex.Message}");
    throw;
}
}

```

- Para API obter detalhes, consulte [AssociateAddress](#) em AWS SDK for .NET API Referência.

Bash

AWS CLI com script Bash

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```

#####
# 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."
    fi
}
```

```

    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
}

```

As funções utilitárias usadas neste exemplo.

```

#####
# 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
}
```

- Para API obter detalhes, consulte [AssociateAddress](#) na Referência de AWS CLI Comandos.

C++

SDK para C++

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

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

//! Associate an Elastic IP address with an EC2 instance.
/*!
 \param instanceId: An EC2 instance ID.
 \param allocationId: An Elastic IP allocation ID.
 \param[out] associationID: String to receive the association ID.
```

```

\param clientConfiguration: AWS client configuration.
\return bool: True if the address was associated with the instance; otherwise,
false.
*/
bool AwsDoc::EC2::associateAddress(const Aws::String &instanceId, const
    Aws::String &allocationId,
                                   Aws::String &associationID,
                                   const Aws::Client::ClientConfiguration
    &clientConfiguration) {
    Aws::EC2::EC2Client ec2Client(clientConfiguration);

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

    Aws::EC2::Model::AssociateAddressOutcome outcome =
    ec2Client.AssociateAddress(request);

    if (!outcome.IsSuccess()) {
        std::cerr << "Failed to associate address " << allocationId <<
            " with instance " << instanceId << ": " <<
            outcome.GetError().GetMessage() << std::endl;
    } else {
        std::cout << "Successfully associated address " << allocationId <<
            " with instance " << instanceId << std::endl;
        associationID = outcome.GetResult().GetAssociationId();
    }

    return outcome.IsSuccess();
}

```

- Para API obter detalhes, consulte [AssociateAddress](#) em AWS SDK for C++ API Referência.

CLI

AWS CLI

Para associar endereços IP elásticos em EC2 -Classic

Este exemplo associa um endereço IP elástico a uma instância em EC2 -Classic. Se o comando for bem-sucedido, nenhuma saída será retornada.

Comando:

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

Para associar um endereço IP elástico em EC2 - VPC

Este exemplo associa um endereço IP elástico a uma instância em umVPC.

Comando:

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

Saída:

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

Este exemplo associa um endereço IP elástico a uma interface de rede.

Comando:

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

Este exemplo associa um IP elástico a um endereço IP privado associado a uma interface de rede.

Comando:

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

- Para API obter detalhes, consulte [AssociateAddress](#) na Referência de AWS CLI Comandos.

Java

SDK para Java 2.x

 Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
/**
 * Associates an Elastic IP address with an EC2 instance asynchronously.
 *
 * @param instanceId the ID of the EC2 instance to associate the Elastic
IP address with
 * @param allocationId the allocation ID of the Elastic IP address to
associate
 * @return a {@link CompletableFuture} that completes with the association ID
when the operation is successful,
 *         or throws a {@link RuntimeException} if the operation fails
 */
public CompletableFuture<String> associateAddressAsync(String instanceId,
String allocationId) {
    AssociateAddressRequest associateRequest =
AssociateAddressRequest.builder()
        .instanceId(instanceId)
        .allocationId(allocationId)
        .build();

    CompletableFuture<AssociateAddressResponse> responseFuture =
getAsyncClient().associateAddress(associateRequest);
    return responseFuture.thenApply(response -> {
        if (response.associationId() != null) {
            return response.associationId();
        } else {
            throw new RuntimeException("Association ID is null after
associating address.");
        }
    }).whenComplete((result, ex) -> {
        if (ex != null) {
            throw new RuntimeException("Failed to associate address", ex);
        }
    });
}
```

```
    });
  }
```

- Para API obter detalhes, consulte [AssociateAddress](#) em AWS SDK for Java 2.x API Referência.

JavaScript

SDK para JavaScript (v3)

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
import { AssociateAddressCommand, EC2Client } from "@aws-sdk/client-ec2";

/**
 * Associates an Elastic IP address, or carrier IP address (for instances that
 * are in subnets in Wavelength Zones)
 * with an instance or a network interface.
 * @param {{ instanceId: string, allocationId: string }} options
 */
export const main = async ({ instanceId, allocationId }) => {
  const client = new EC2Client({});
  const command = new AssociateAddressCommand({
    // You need to allocate an Elastic IP address before associating it with an
    // instance.
    // You can do that with the AllocateAddressCommand.
    AllocationId: allocationId,
    // You need to create an EC2 instance before an IP address can be associated
    // with it.
    // You can do that with the RunInstancesCommand.
    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 ${AssociationId}.`,
    );
} catch (caught) {
    if (
        caught instanceof Error &&
        caught.name === "InvalidAllocationID.NotFound"
    ) {
        console.warn(
            `${caught.message}. Did you provide the ID of a valid Elastic IP address
            AllocationId?`,
        );
    } else {
        throw caught;
    }
}
};

```

- Para API obter detalhes, consulte [AssociateAddress](#) em AWS SDK for JavaScript API Referência.

Kotlin

SDK para Kotlin

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```

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  
    }  
}
```

- Para API obter detalhes, consulte a [AssociateAddress](#) referência AWS SDKdo Kotlin API.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo associa o endereço IP elástico especificado à instância especificada em umVPC.

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

Saída:

```
eipassoc-12345678
```


Exemplo 2: Este exemplo associa o endereço IP elástico especificado à instância especificada em EC2 -Classic.

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

- Para API obter detalhes, consulte [AssociateAddress](#) em Referência de AWS Tools for PowerShell cmdlet.

Python

SDK para Python (Boto3)

 Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

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

    class ElasticIp:
        """Represents an Elastic IP and its associated instance."""

        def __init__(
            self, allocation_id: str, public_ip: str, instance_id: Optional[str]
            = None
        ) -> None:
            """
            Initializes the ElasticIp object.

            :param allocation_id: The allocation ID of the Elastic IP.
            :param public_ip: The public IP address of the Elastic IP.
            :param instance_id: The ID of the associated EC2 instance, if any.
            """
            self.allocation_id = allocation_id
            self.public_ip = public_ip
            self.instance_id = instance_id

        def __init__(self, ec2_client: Any) -> None:
            """
            Initializes the ElasticIpWrapper with an EC2 client.

            :param ec2_client: A Boto3 Amazon EC2 client. This client provides low-
            level
                                access to AWS EC2 services.
            """
            self.ec2_client = ec2_client
            self.elastic_ips: List[ElasticIpWrapper.ElasticIp] = []
```

```
@classmethod
def from_client(cls) -> "ElasticIpWrapper":
    """
    Creates an ElasticIpWrapper instance with a default EC2 client.

    :return: An instance of ElasticIpWrapper initialized with the default EC2
client.
    """
    ec2_client = boto3.client("ec2")
    return cls(ec2_client)

def associate(
    self, allocation_id: str, instance_id: str
) -> Union[Dict[str, Any], None]:
    """
    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 allocation_id: The allocation ID of the Elastic IP.
    :param instance_id: The ID of the Amazon EC2 instance.
    :return: A response that contains the ID of the association, or None if
no Elastic IP is found.
    :raises ClientError: If the association fails, such as when the instance
ID is not found.
    """
    elastic_ip = self.get_elastic_ip_by_allocation(self.elastic_ips,
allocation_id)
    if elastic_ip is None:
        logger.info(f"No Elastic IP found with allocation ID
{allocation_id}.")
        return None

    try:
        response = self.ec2_client.associate_address(
            AllocationId=allocation_id, InstanceId=instance_id
        )
        elastic_ip.instance_id = (
            instance_id # Track the instance associated with this Elastic
IP.
```

```
    )
    except ClientError as err:
        if err.response["Error"]["Code"] == "InvalidInstanceID.NotFound":
            logger.error(
                f"Failed to associate Elastic IP {allocation_id} with
{instance_id} "
                "because the specified instance ID does not exist or has not
propagated fully. "
                "Verify the instance ID and try again, or wait a few moments
before attempting to "
                "associate the Elastic IP address."
            )
            raise
    return response
```

- Para API obter detalhes, consulte a [AssociateAddress](#) Referência AWS SDK do Python (Boto3). API

Ruby

SDK para Ruby

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
# 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 association 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
  )
  response.association_id
rescue StandardError => e
  puts "Error associating Elastic IP address with instance: #{e.message}"
  'Error'
end

```

- Para API obter detalhes, consulte [AssociateAddress](#) em AWS SDK for Ruby API Referência.

Rust

SDK para Rust

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```

pub async fn associate_ip_address(
  &self,
  allocation_id: &str,
  instance_id: &str,
) -> Result<AssociateAddressOutput, EC2Error> {

```

```

    let response = self
      .client
      .associate_address()
      .allocation_id(allocation_id)
      .instance_id(instance_id)
      .send()
      .await?;
    Ok(response)
  }

```

- Para API obter detalhes, consulte [AssociateAddress](#) AWS SDK API referência do Rust.

SAP ABAP

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```

TRY.
    oo_result = lo_ec2->associateaddress(
is returned for testing purposes. " oo_result
        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.

```

- Para API obter detalhes, consulte [AssociateAddress AWS SDK para SAP ABAP](#) API referência.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use `AssociateDhcpOptions` com um CLI

Os exemplos de código a seguir mostram como usar o `AssociateDhcpOptions`.

CLI

AWS CLI

Para associar um conjunto de DHCP opções ao seu VPC

Este exemplo associa o conjunto DHCP de opções especificado ao especificadoVPC. Se o comando for bem-sucedido, nenhuma saída será retornada.

Comando:

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

Para associar as DHCP opções padrão definidas ao seu VPC

Este exemplo associa o conjunto DHCP de opções padrão ao especificadoVPC. Se o comando for bem-sucedido, nenhuma saída será retornada.

Comando:

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

- Para API obter detalhes, consulte [AssociateDhcpOptions](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo associa o conjunto de DHCP opções especificado ao especificadoVPC.


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

Exemplo 2: Este exemplo associa o conjunto de DHCP opções padrão ao especificado VPC.

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

- Para API obter detalhes, consulte [AssociateDhcpOptions](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **AssociateRouteTable** com um CLI

Os exemplos de código a seguir mostram como usar o AssociateRouteTable.

CLI

AWS CLI

Para associar uma tabela de rotas a uma sub-rede

Este exemplo associa a tabela de rotas especificada à sub-rede especificada.

Comando:

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

Saída:

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

- Para API obter detalhes, consulte [AssociateRouteTable](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo associa a tabela de rotas especificada à sub-rede especificada.

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

Saída:

```
rtbassoc-12345678
```

- Para API obter detalhes, consulte [AssociateRouteTable](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **AttachInternetGateway** com um CLI

Os exemplos de código a seguir mostram como usar o `AttachInternetGateway`.

CLI

AWS CLI

Para conectar um gateway de internet ao seu VPC

O `attach-internet-gateway` exemplo a seguir anexa o gateway de internet especificado ao específico VPC.

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

Este comando não produz saída.

Para obter mais informações, consulte [Gateways de Internet](#) no Guia do VPC usuário da Amazon.

- Para API obter detalhes, consulte [AttachInternetGateway](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo conecta o gateway de Internet especificado ao especificado VPC.

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

Exemplo 2: Este exemplo cria um VPC e um gateway de Internet e, em seguida, conecta o gateway de Internet ao VPC.

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

- Para API obter detalhes, consulte [AttachInternetGateway](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **AttachNetworkInterface** com um CLI

Os exemplos de código a seguir mostram como usar o `AttachNetworkInterface`.

CLI

AWS CLI

Exemplo 1: Para conectar uma interface de rede a uma instância

O `attach-network-interface` exemplo a seguir anexa a interface de rede especificada à instância especificada.

```
aws ec2 attach-network-interface \  
  --network-interface-id eni-0dc56a8d4640ad10a \  
  --instance-id i-01234567890123456 \  
  --subnet-id subnet-01234567890123456 \  
  --security-groups sg-01234567890123456 \  
  --tags Key=Value \  
  --dry-run
```

```
--instance-id i-1234567890abcdef0 \  
--device-index 1
```

Saída:

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

Para obter mais informações, consulte [Interfaces de rede elásticas](#) no Guia EC2 do usuário da Amazon.

Exemplo 2: Para conectar uma interface de rede a uma instância com várias placas de rede

O `attach-network-interface` exemplo a seguir anexa a interface de rede especificada à instância e à placa de rede especificadas.

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

Saída:

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

Para obter mais informações, consulte [Interfaces de rede elásticas](#) no Guia EC2 do usuário da Amazon.

- Para API obter detalhes, consulte [AttachNetworkInterface](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo anexa a interface de rede especificada à instância especificada.

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

Saída:

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

- Para API obter detalhes, consulte [AttachNetworkInterface](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **AttachVolume** com um CLI

Os exemplos de código a seguir mostram como usar o `AttachVolume`.

CLI

AWS CLI

Para anexar um volume a uma instância

Este exemplo de comando anexa um volume (`vol-1234567890abcdef0`) a uma instância (`i-01474ef662b89480`) como `/dev/sdf`.

Comando:

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

Saída:

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

```
"Device": "/dev/sdf"
}
```

- Para API obter detalhes, consulte [AttachVolume](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo anexa o volume especificado à instância especificada e o expõe com o nome do dispositivo especificado.

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

Saída:

```
AttachTime      : 12/22/2015 1:53:58 AM
DeleteOnTermination : False
Device          : /dev/sdh
InstanceId      : i-1a2b3c4d
State           : attaching
VolumeId       : vol-12345678
```

- Para API obter detalhes, consulte [AttachVolume](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **AttachVpnGateway** com um CLI

Os exemplos de código a seguir mostram como usar o `AttachVpnGateway`.

CLI

AWS CLI

Para conectar um gateway privado virtual ao seu VPC

O `attach-vpn-gateway` exemplo a seguir anexa o gateway privado virtual especificado ao especificadoVPC.

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

Saída:

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

- Para API obter detalhes, consulte [AttachVpnGateway](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo anexa o gateway privado virtual especificado ao especificadoVPC.

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

Saída:

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

- Para API obter detalhes, consulte [AttachVpnGateway](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use `AuthorizeSecurityGroupEgress` com um CLI

Os exemplos de código a seguir mostram como usar o `AuthorizeSecurityGroupEgress`.

CLI

AWS CLI

Para adicionar uma regra que permita tráfego de saída para um intervalo de endereços específico

Este exemplo de comando adiciona uma regra que concede acesso aos intervalos de endereços especificados na TCP porta 80.

Comando (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}}]'
```

Comando (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}}]
```

Para adicionar uma regra que permita tráfego de saída para um grupo de segurança específico

Este exemplo de comando adiciona uma regra que concede acesso ao grupo de segurança especificado na TCP porta 80.

Comando (Linux):

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

Comando (Windows):

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

- Para API obter detalhes, consulte [AuthorizeSecurityGroupEgress](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo define uma regra de saída para o grupo de segurança especificado para EC2 -VPC. A regra concede acesso ao intervalo de endereços IP especificado na TCP porta 80. A sintaxe usada neste exemplo requer a PowerShell versão 3 ou superior.

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

Exemplo 2: Com a PowerShell versão 2, você deve usar `New-Object` para criar o `IpPermission` objeto.

```
$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
```

Exemplo 3: Este exemplo concede acesso ao grupo de segurança de origem especificado na TCP porta 80.

```
$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 } )
```

- Para API obter detalhes, consulte [AuthorizeSecurityGroupEgress](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use `AuthorizeSecurityGroupIngress` com um AWS SDK ou CLI

Os exemplos de código a seguir mostram como usar o `AuthorizeSecurityGroupIngress`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação no contexto no seguinte exemplo de código:

- [Aprender os conceitos básicos](#)

.NET

AWS SDK for .NET

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
/// <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)
{
    try
    {
        // 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;
    }
    catch (AmazonEC2Exception ec2Exception)
    {
        if (ec2Exception.ErrorCode == "InvalidPermission.Duplicate")
        {
            _logger.LogError(
                $"The ingress rule already exists. {ec2Exception.Message}");
        }

        throw;
    }
    catch (Exception ex)
    {
        _logger.LogError(
            $"An error occurred while authorizing ingress.: {ex.Message}");
        throw;
    }
}

/// <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();
}

```

- Para API obter detalhes, consulte [AuthorizeSecurityGroupIngress](#) em AWS SDK for .NET APIReferência.

Bash

AWS CLI com script Bash

 Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
#####
# 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 getopt 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
}

```

As funções utilitárias usadas neste exemplo.

```

#####
# 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
}
```

- Para API obter detalhes, consulte [AuthorizeSecurityGroupIngress](#) na Referência de AWS CLI Comandos.

C++

SDK para C++

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```

//! Authorize ingress to an Amazon Elastic Compute Cloud (Amazon EC2) group.
/*!
  \param groupID: The EC2 group ID.
  \param clientConfiguration: The ClientConfiguration object.
  \return bool: True if the operation was successful, false otherwise.
*/
bool
AwsDoc::EC2::authorizeSecurityGroupIngress(const Aws::String &groupID,
                                           const Aws::Client::ClientConfiguration
                                           &clientConfiguration) {
    Aws::EC2::EC2Client ec2Client(clientConfiguration);
    Aws::EC2::Model::AuthorizeSecurityGroupIngressRequest
authorizeSecurityGroupIngressRequest;
    authorizeSecurityGroupIngressRequest.SetGroupId(groupID);
    buildSampleIngressRule(authorizeSecurityGroupIngressRequest);

    Aws::EC2::Model::AuthorizeSecurityGroupIngressOutcome
authorizeSecurityGroupIngressOutcome =

ec2Client.AuthorizeSecurityGroupIngress(authorizeSecurityGroupIngressRequest);

    if (authorizeSecurityGroupIngressOutcome.IsSuccess()) {
        std::cout << "Successfully authorized security group ingress." <<
std::endl;
    } else {
        std::cerr << "Error authorizing security group ingress: "
        << authorizeSecurityGroupIngressOutcome.GetError().GetMessage()
<< std::endl;
    }

    return authorizeSecurityGroupIngressOutcome.IsSuccess();
}

```

Função utilitária para criar uma regra de entrada.

```

//! Build a sample ingress rule.
/*!
  \param authorize_request: An 'AuthorizeSecurityGroupIngressRequest' instance.
  \return void:
*/
void buildSampleIngressRule(

```



```
    Aws::EC2::Model::AuthorizeSecurityGroupIngressRequest &authorize_request)
{
    Aws::String ingressIPRange = "203.0.113.0/24"; // Configure this for your
allowed IP range.
    Aws::EC2::Model::IpRange ip_range;
    ip_range.SetCidrIp(ingressIPRange);

    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);
}
```

- Para API obter detalhes, consulte [AuthorizeSecurityGroupIngress](#) em AWS SDK for C++ APIReferência.

CLI

AWS CLI

Exemplo 1: Para adicionar uma regra que permita tráfego de entrada SSH

O `authorize-security-group-ingress` exemplo a seguir adiciona uma regra que permite tráfego de entrada na TCP porta 22 (SSH).

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

Saída:

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

Exemplo 2: Para adicionar uma regra que permita HTTP tráfego de entrada de outro grupo de segurança

O `authorize-security-group-ingress` exemplo a seguir adiciona uma regra que permite acesso de entrada na TCP porta 80 do grupo `sg-1a2b3c4d` de segurança de origem. O grupo de origem deve estar no mesmo grupo VPC ou em um peer VPC (requer uma conexão de VPC peering). O tráfego de entrada é permitido com base nos endereços IP privados das instâncias associadas ao grupo de segurança de origem (e não ao endereço IP público ou ao endereço IP elástico).

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

Saída:

```
{
  "Return": true,
  "SecurityGroupRules": [
    {
      "SecurityGroupId": "sgr-01f4be99110f638a7",
```

```

    "GroupId": "sg-1234567890abcdef0",
    "GroupOwnerId": "123456789012",
    "IsEgress": false,
    "IpProtocol": "tcp",
    "FromPort": 80,
    "ToPort": 80,
    "ReferencedGroupInfo": {
      "GroupId": "sg-1a2b3c4d",
      "UserId": "123456789012"
    }
  }
]
}

```

Exemplo 3: adicionar várias regras na mesma chamada

O `authorize-security-group-ingress` exemplo a seguir usa o `ip-permissions` parâmetro para adicionar duas regras de entrada, uma que habilita o acesso de entrada na TCP porta 3389 (RDP) e a outra que habilita ping/. ICMP

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

Saída:

```

{
  "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"
    }
]
}

```

Exemplo 4: Para adicionar uma regra para ICMP tráfego

O `authorize-security-group-ingress` exemplo a seguir usa o `ip-permissions` parâmetro para adicionar uma regra de entrada que permite a ICMP mensagem `Destination Unreachable: Fragmentation Needed and Don't Fragment was Set` (Tipo 3, Código 4) de qualquer lugar.

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

Saída:

```

{
  "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"
    }
  ]
}

```

Exemplo 5: Para adicionar uma regra para IPv6 tráfego

O `authorize-security-group-ingress` exemplo a seguir usa o `ip-permissions` parâmetro para adicionar uma regra de entrada que permite SSH acesso (porta 22) a partir do IPv6 intervalo `2001:db8:1234:1a00::/64`.

```
aws ec2 authorize-security-group-ingress --group-id sg-1234567890abcdef0 --ip-permissions
=tcp, =22, =22, Ipv6Ranges= "[{6=2001:db 8:1234:1 a00: :/64 IpProtocol}]" FromPort ToPort
CidrIpv
```

Saída:

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

Exemplo 6: Para adicionar uma regra para ICMPv6 tráfego

O `authorize-security-group-ingress` exemplo a seguir usa o `ip-permissions` parâmetro para adicionar uma regra de entrada que permite ICMPv6 tráfego de qualquer lugar.

```
aws ec2 authorize-security-group-ingress --group-id sg-1234567890abcdef0 --ip-permissions
=icmpv6, Ipv6Ranges= "[{6=: :/0}]" IpProtocol CidrIpv
```

Saída:

```
{
  "Return": true,
  "SecurityGroupRules": [
    {
      "SecurityGroupRuleId": "sgr-04b612d9363ab6327",
      "GroupId": "sg-1234567890abcdef0",
      "GroupOwnerId": "123456789012",
      "IsEgress": false,
      "IpProtocol": "icmpv6",

```

```

        "FromPort": -1,
        "ToPort": -1,
        "CidrIpv6": ":::/0"
    }
]
}

```

Exemplo 7: adicionar uma regra com uma descrição

O `authorize-security-group-ingress` exemplo a seguir usa o `ip-permissions` parâmetro para adicionar uma regra de entrada que permite o RDP tráfego do intervalo de IPv4 endereços especificado. A regra inclui uma descrição para ajudar a identificá-lo posteriormente.

```
aws ec2 authorize-security-group-ingress --group-id sg-1234567890abcdef0 --ip-permissions IpProtocol=tcp,=3389,=3389,="[=203.0.113.0/24, Descrição=' acesso do escritório de Nova York']" FromPort ToPort IpRanges CidrIp RDP
```

Saída:

```

{
  "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"
    }
  ]
}

```

Exemplo 8: adicionar uma regra de entrada que use uma lista de prefixos

O `authorize-security-group-ingress` exemplo a seguir usa o `ip-permissions` parâmetro para adicionar uma regra de entrada que permite todo o tráfego para os CIDR intervalos na lista de prefixos especificada.

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

Saída:


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

Para obter mais informações, consulte [Grupos de segurança](#) no Guia VPC do usuário da Amazon.

- Para API obter detalhes, consulte [AuthorizeSecurityGroupIngress](#) na Referência de AWS CLI Comandos.

Java

SDK para Java 2.x

 Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
/**
 * Creates a new security group asynchronously with the specified group name,
 * description, and VPC ID. It also
```

```
    * authorizes inbound traffic on ports 80 and 22 from the specified IP
address.
    *
    * @param groupName    the name of the security group to create
    * @param groupDesc    the description of the security group
    * @param vpcId        the ID of the VPC in which to create the security
group
    * @param myIpAddress  the IP address from which to allow inbound traffic
(e.g., "192.168.1.1/0" to allow traffic from
    *                      any IP address in the 192.168.1.0/24 subnet)
    * @return a CompletableFuture that, when completed, returns the ID of the
created security group
    * @throws RuntimeException if there was a failure creating the security
group or authorizing the inbound traffic
    */
    public CompletableFuture<String> createSecurityGroupAsync(String groupName,
String groupDesc, String vpcId, String myIpAddress) {
        CreateSecurityGroupRequest createRequest =
CreateSecurityGroupRequest.builder()
            .groupName(groupName)
            .description(groupDesc)
            .vpcId(vpcId)
            .build();

        return getAsyncClient().createSecurityGroup(createRequest)
            .thenCompose(createResponse -> {
                String groupId = createResponse.groupId();
                IpRange ipRange = IpRange.builder()
                    .cidrIp(myIpAddress + "/32")
                    .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();

        return
getAsyncClient().authorizeSecurityGroupIngress(authRequest)
        .thenApply(authResponse -> groupId);
    })
    .whenComplete((result, exception) -> {
        if (exception != null) {
            if (exception instanceof CompletionException &&
exception.getCause() instanceof Ec2Exception) {
                throw (Ec2Exception) exception.getCause();
            } else {
                throw new RuntimeException("Failed to create security
group: " + exception.getMessage(), exception);
            }
        }
    });
}
```

- Para API obter detalhes, consulte [AuthorizeSecurityGroupIngress](#) em AWS SDK for Java 2.x API Referência.

JavaScript

SDK para JavaScript (v3)

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
import {
    AuthorizeSecurityGroupIngressCommand,
    EC2Client,
```

```

} from "@aws-sdk/client-ec2";

/**
 * Adds the specified inbound (ingress) rules to a security group.
 * @param {{ groupId: string, ipAddress: string }} options
 */
export const main = async ({ groupId, ipAddress }) => {
  const client = new EC2Client({});
  const command = new AuthorizeSecurityGroupIngressCommand({
    // Use a group ID from the AWS console or
    // the DescribeSecurityGroupsCommand.
    GroupId: groupId,
    IpPermissions: [
      {
        IpProtocol: "tcp",
        FromPort: 22,
        ToPort: 22,
        // 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: `${ipAddress}/32` }],
      },
    ],
  });

  try {
    const { SecurityGroupRules } = await client.send(command);
    console.log(JSON.stringify(SecurityGroupRules, null, 2));
  } catch (caught) {
    if (caught instanceof Error && caught.name === "InvalidGroupId.Malformed") {
      console.warn(`${caught.message}. Please provide a valid GroupId.`);
    } else {
      throw caught;
    }
  }
};

```

- Para API obter detalhes, consulte [AuthorizeSecurityGroupIngress](#) em AWS SDK for JavaScript API Referência.

Kotlin

SDK para Kotlin

 Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
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
}
}

```

- Para API obter detalhes, consulte a [AuthorizeSecurityGroupIngress](#) referência AWS SDK do Kotlin API.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo define regras de entrada para um grupo de segurança para EC2 -VPC. Essas regras concedem acesso a um endereço IP específico para SSH (porta 22) e RDC (porta 3389). Observe que você deve identificar grupos de segurança para EC2 VPC usar o ID do grupo de segurança e não o nome do grupo de segurança. A sintaxe usada neste exemplo requer a PowerShell versão 3 ou superior.

```

$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 )

```

Exemplo 2: Com a PowerShell versão 2, você deve usar `New-Object` para criar os `IpPermission` objetos.

```

$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 )
```

Exemplo 3: Este exemplo define regras de entrada para um grupo de segurança para EC2 - Classic. Essas regras concedem acesso a um endereço IP específico para SSH (porta 22) e RDC (porta 3389). A sintaxe usada neste exemplo requer a PowerShell versão 3 ou superior.

```
$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 )
```

Exemplo 4: Com a PowerShell versão 2, você deve usar New-Object para criar os IpPermission objetos.

```
$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 )
```

Exemplo 5: Este exemplo concede acesso à TCP porta 8081 do grupo de segurança de origem especificado (sg-1a2b3c4d) ao grupo de segurança especificado (sg-12345678).

```
$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 } )
```

Exemplo 6: Este exemplo adiciona o CIDR 5.5.5.5/32 às regras de entrada do grupo de segurança sg-1234abcd para o tráfego da porta 22 com uma descrição. TCP

```
$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
```

- Para API obter detalhes, consulte [AuthorizeSecurityGroupIngress](#) em Referência de AWS Tools for PowerShell cmdlet.

Python

SDK para Python (Boto3)

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

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

```
def __init__(self, ec2_client: boto3.client, security_group: Optional[str] =
None):
    """
    Initializes the SecurityGroupWrapper with an EC2 client and an optional
    security group ID.

    :param ec2_client: A Boto3 Amazon EC2 client. This client provides low-
level
                        access to AWS EC2 services.
    :param security_group: The ID of a security group to manage. This is a
high-level identifier
                        that represents the security group.
    """
    self.ec2_client = ec2_client
    self.security_group = security_group

    @classmethod
    def from_client(cls) -> "SecurityGroupWrapper":
        """
        Creates a SecurityGroupWrapper instance with a default EC2 client.

        :return: An instance of SecurityGroupWrapper initialized with the default
EC2 client.
        """
        ec2_client = boto3.client("ec2")
        return cls(ec2_client)

    def authorize_ingress(self, ssh_ingress_ip: str) -> Optional[Dict[str, Any]]:
        """
        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, or
None if no security group is set.
        :raise Handles AWS SDK service-level ClientError, with special handling
for ResourceAlreadyExists
        """
        if self.security_group is None:
            logger.info("No security group to update.")
```

```
        return None

    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.ec2_client.authorize_security_group_ingress(
            GroupId=self.security_group, IpPermissions=ip_permissions
        )
    except ClientError as err:
        if err.response["Error"]["Code"] == "InvalidPermission.Duplicate":
            logger.error(
                f"The SSH ingress rule for IP {ssh_ingress_ip} already exists"
                f"in security group '{self.security_group}'."
            )
            raise
        else:
            return response
```

- Para API obter detalhes, consulte a [AuthorizeSecurityGroupIngress](#) Referência AWS SDK do Python (Boto3). API

Rust

SDK para Rust

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).


```

/// Add an ingress rule to a security group explicitly allowing IPv4 address
/// as {ip}/32 over TCP port 22.
pub async fn authorize_security_group_ssh_ingress(
    &self,
    group_id: &str,
    ingress_ips: Vec<Ipv4Addr>,
) -> Result<(), EC2Error> {
    tracing::info!("Authorizing ingress for security group {group_id}");
    self.client
        .authorize_security_group_ingress()
        .group_id(group_id)
        .set_ip_permissions(Some(
            ingress_ips
                .into_iter()
                .map(|ip| {
                    IpPermission::builder()
                        .ip_protocol("tcp")
                        .from_port(22)
                        .to_port(22)
                        .ip_ranges(IpRange::builder().cidr_ip(format!(
                            "{ip}/32"))).build())
                })
                .build()
                .collect(),
        ))
        .send()
        .await?;
    Ok(())
}

```

- Para API obter detalhes, consulte [AuthorizeSecurityGroupIngress](#) a AWS SDK API referência do Rust.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use `CancelCapacityReservation` com um CLI

Os exemplos de código a seguir mostram como usar o `CancelCapacityReservation`.

CLI

AWS CLI

Para cancelar uma reserva de capacidade

O `cancel-capacity-reservation` exemplo a seguir cancela a reserva de capacidade especificada.

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

Saída:

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

Para obter mais informações, consulte [Cancelamento de uma reserva de capacidade](#) no Guia do usuário do Amazon Elastic Compute Cloud para instâncias Linux.

- Para API obter detalhes, consulte [CancelCapacityReservation](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo cancela a reserva de capacidade `cr-0c1f2345db6f7cdba`

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

Saída:

```
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
```

- Para API obter detalhes, consulte [CancelCapacityReservation](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **CancelImportTask** com um CLI

Os exemplos de código a seguir mostram como usar o `CancelImportTask`.

CLI

AWS CLI

Para cancelar uma tarefa de importação

O `cancel-import-task` exemplo a seguir cancela a tarefa de importação de imagem especificada.

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

Saída:

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

- Para API obter detalhes, consulte [CancelImportTask](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo cancela a tarefa de importação especificada (importação de instantâneo ou imagem). Se necessário, um motivo pode ser fornecido usando o `-CancelReason` parâmetro.

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

- Para API obter detalhes, consulte [CancelImportTask](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use `CancelSpotFleetRequests` com um CLI

Os exemplos de código a seguir mostram como usar o `CancelSpotFleetRequests`.

CLI

AWS CLI

Exemplo 1: Para cancelar uma solicitação de frota spot e encerrar as instâncias associadas

O `cancel-spot-fleet-requests` exemplo a seguir cancela uma solicitação de frota spot e encerra as instâncias sob demanda e as instâncias spot associadas.

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

Saída:

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

Para obter mais informações, consulte [Cancelar uma solicitação de frota spot](#) no Guia do usuário do Amazon Elastic Compute Cloud para instâncias Linux.

Exemplo 2: Cancelar uma solicitação de frota spot sem encerrar as instâncias associadas

O `cancel-spot-fleet-requests` exemplo a seguir cancela uma solicitação de frota spot sem encerrar as instâncias sob demanda e as instâncias spot associadas.

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

Saída:

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

Para obter mais informações, consulte [Cancelar uma solicitação de frota spot](#) no Guia do usuário do Amazon Elastic Compute Cloud para instâncias Linux.

- Para API obter detalhes, consulte [CancelSpotFleetRequests](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo cancela a solicitação de frota spot especificada e encerra as instâncias spot associadas.

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

Exemplo 2: Este exemplo cancela a solicitação de frota spot especificada sem encerrar as instâncias spot associadas.

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

- Para API obter detalhes, consulte [CancelSpotFleetRequests](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use `CancelSpotInstanceRequests` com um CLI

Os exemplos de código a seguir mostram como usar o `CancelSpotInstanceRequests`.

CLI

AWS CLI

Para cancelar solicitações de Instância Spot

Este exemplo de comando cancela uma solicitação de instância spot.

Comando:

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

Saída:

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

- Para API obter detalhes, consulte [CancelSpotInstanceRequests](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo cancela a solicitação de instância spot especificada.

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

Saída:

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

- Para API obter detalhes, consulte [CancelSpotInstanceRequests](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **ConfirmProductInstance** com um CLI

Os exemplos de código a seguir mostram como usar o `ConfirmProductInstance`.

CLI

AWS CLI

Para confirmar a instância do produto

Esse exemplo determina se o código do produto especificado está associado à instância especificada.

Comando:

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

Saída:

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

- Para API obter detalhes, consulte [ConfirmProductInstance](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Esse exemplo determina se o código do produto especificado está associado à instância especificada.

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

- Para API obter detalhes, consulte [ConfirmProductInstance](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **CopyImage** com um CLI

Os exemplos de código a seguir mostram como usar o CopyImage.

CLI

AWS CLI

Exemplo 1: Para copiar um AMI para outra região

O comando de `copy-image` exemplo a seguir copia o especificado AMI da `us-west-2` Região para a `us-east-1` Região e adiciona uma breve descrição.

```
aws ec2 copy-image \  
  --region us-east-1 \  
  --name ami-name \  
  --source-image-id ami-id \  
  --description description
```



```
--source-region us-west-2 \  
--source-image-id ami-066877671789bd71b \  
--description "This is my copied image."
```

Saída:

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

Para obter mais informações, consulte [Copiar um AMI](#) no Guia EC2 do usuário da Amazon.

Exemplo 2: Para copiar um AMI para outra região e criptografar o instantâneo de backup

O `copy-image` comando a seguir copia o especificado AMI da `us-west-2` região para a região atual e criptografa o instantâneo de backup usando a chave especificada. KMS

```
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
```

Saída:

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

Para obter mais informações, consulte [Copiar um AMI](#) no Guia EC2 do usuário da Amazon.

Exemplo 3: Para incluir suas AMI tags definidas pelo usuário ao copiar um AMI

O `copy-image` comando a seguir usa o `--copy-image-tags` parâmetro para copiar suas AMI tags definidas pelo usuário ao copiar o. AMI

```
aws ec2 copy-image \  
  --region us-east-1 \  
  --name ami-name \  
  --source-region us-west-2 \  
  --copy-image-tags
```

```
--source-image-id ami-066877671789bd71b \  
--description "This is my copied image."  
--copy-image-tags
```

Saída:

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

Para obter mais informações, consulte [Copiar um AMI](#) no Guia EC2 do usuário da Amazon.

- Para API obter detalhes, consulte [CopyImage](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo copia o especificado AMI na região “UE (Irlanda)” para a região “Oeste dos EUA (Oregon)”. Se -Region não for especificada, a região padrão atual será usada como a região de destino.

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

Saída:

```
ami-87654321
```

- Para API obter detalhes, consulte [CopyImage](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **CopySnapshot** com um CLI

Os exemplos de código a seguir mostram como usar o CopySnapshot.

CLI

AWS CLI

Exemplo 1: Para copiar um instantâneo para outra região

O comando de `copy-snapshot` exemplo a seguir copia o instantâneo especificado da `us-west-2` região para a `us-east-1` região e adiciona uma breve descrição.

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

Saída:

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

Para obter mais informações, consulte [Copiar um EBS snapshot da Amazon](#) no Guia do EC2 usuário da Amazon.

Exemplo 2: Para copiar um instantâneo não criptografado e criptografar o novo instantâneo

O `copy-snapshot` comando a seguir copia o instantâneo não criptografado especificado da `us-west-2` região para a região atual e criptografa o novo instantâneo usando a chave especificada. KMS

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

Saída:

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

Para obter mais informações, consulte [Copiar um EBS snapshot da Amazon](#) no Guia do EC2 usuário da Amazon.

- Para API obter detalhes, consulte [CopySnapshot](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo copia o snapshot especificado da região da UE (Irlanda) para a região Oeste dos EUA (Oregon).

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

Exemplo 2: Se você definir uma região padrão e omitir o parâmetro Região, a região de destino padrão será a região padrão.

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

- Para API obter detalhes, consulte [CopySnapshot](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **CreateCapacityReservation** com um CLI

Os exemplos de código a seguir mostram como usar o `CreateCapacityReservation`.

CLI

AWS CLI

Exemplo 1: Para criar uma reserva de capacidade

O `create-capacity-reservation` exemplo a seguir cria uma reserva de capacidade na zona de eu-west-1a disponibilidade, na qual você pode executar três `t2.medium` instâncias

executando um sistema operacional Linux/Unix. Por padrão, a reserva de capacidade é criada com critérios de correspondência de instâncias abertas e sem suporte para armazenamento temporário, e permanece ativa até que você a cancele manualmente.

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

Saída:

```
{  
  "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"  
  }  
}
```

Exemplo 2: Para criar uma reserva de capacidade que termine automaticamente em uma data/hora especificada

O `create-capacity-reservation` exemplo a seguir cria uma reserva de capacidade na zona de eu-west-1a disponibilidade, na qual você pode executar três `m5.large` instâncias executando um sistema operacional Linux/Unix. Essa reserva de capacidade termina automaticamente em 31/08/2019 às 23:59:59.

```
aws ec2 create-capacity-reservation \  
  --availability-zone eu-west-1a \  
  --instance-type m5.large \  
  --end-date 2019-08-31T23:59:59
```

```
--instance-platform Linux/UNIX \  
--instance-count 3 \  
--end-date-type limited \  
--end-date 2019-08-31T23:59:59Z
```

Saída:

```
{  
  "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"  
  }  
}
```

Exemplo 3: Para criar uma reserva de capacidade que aceite somente inicializações de instâncias específicas

O `create-capacity-reservation` exemplo a seguir cria uma reserva de capacidade que aceita somente lançamentos de instâncias direcionadas.

```
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
```

Saída:

```
{
```

```
"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"
}
```

Para obter mais informações, consulte [Criação de uma reserva de capacidade](#) no Guia do usuário do Amazon Elastic Compute Cloud para instâncias Linux.

- Para API obter detalhes, consulte [CreateCapacityReservation](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo cria uma nova reserva de capacidade com os atributos especificados

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

Saída:

```
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
```

- Para API obter detalhes, consulte [CreateCapacityReservation](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use `CreateCustomerGateway` com um CLI

Os exemplos de código a seguir mostram como usar o `CreateCustomerGateway`.

CLI

AWS CLI

Para criar um gateway do cliente

Este exemplo cria um gateway de cliente com o endereço IP especificado para sua interface externa.

Comando:

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

Saída:

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



```
    "BgpAsn": "65534"  
  }  
}
```

- Para API obter detalhes, consulte [CreateCustomerGateway](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Esse exemplo cria o gateway do cliente especificado.

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

Saída:

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

- Para API obter detalhes, consulte [CreateCustomerGateway](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **CreateDhcpOptions** com um CLI

Os exemplos de código a seguir mostram como usar o `CreateDhcpOptions`.

CLI

AWS CLI

Para criar um conjunto de DHCP opções

O `create-dhcp-options` exemplo a seguir cria um conjunto de DHCP opções que especifica o nome do domínio, os servidores de nomes de domínio e o tipo de BIOS nó da rede.

```
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"
```

Saída:

```
{  
  "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"
  }
}

```

- Para API obter detalhes, consulte [CreateDhcpOptions](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo cria o conjunto especificado de DHCP opções. A sintaxe usada neste exemplo requer a PowerShell versão 3 ou posterior.

```

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

```

Saída:

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

Exemplo 2: Com a PowerShell versão 2, você deve usar `New-Object` para criar cada DHCP opção.

```

$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)

```

Saída:

DhcpConfigurations	DhcpOptionsId	Tags
--------------------	---------------	------

```
-----
{domain-name, domain-name-servers}   dopt-2a3b4c5d   {}
```

- Para API obter detalhes, consulte [CreateDhcpOptions](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **CreateFlowLogs** com um CLI

Os exemplos de código a seguir mostram como usar o `CreateFlowLogs`.

CLI

AWS CLI

Exemplo 1: Para criar um registro de fluxo

O `create-flow-logs` exemplo a seguir cria um registro de fluxo que captura todo o tráfego rejeitado para a interface de rede especificada. Os registros de fluxo são entregues a um grupo de CloudWatch registros no Logs usando as permissões na IAM função especificada.

```
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
```

Saída:

```
{
  "ClientToken": "so0eNA2uSHUN1HI0S2cJ305GuIX1CezaRdGtexample",
  "FlowLogIds": [
    "fl-12345678901234567"
  ],
  "Unsuccessful": []
}
```

Para obter mais informações, consulte [Logs de VPC fluxo](#) no Guia VPC do usuário da Amazon.

Exemplo 2: Para criar um registro de fluxo com um formato personalizado

O `create-flow-logs` exemplo a seguir cria um log de fluxo que captura todo o tráfego para o especificado VPC e entrega os logs de fluxo para um bucket do Amazon S3. O parâmetro `--log-format` especifica um formato personalizado para os registros de log de fluxo. Para executar esse comando no Windows, altere as aspas simples (') para aspas duplas (").

```
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}'
```

Para obter mais informações, consulte [Logs de VPC fluxo](#) no Guia VPC do usuário da Amazon.

Exemplo 3: Para criar um registro de fluxo com um intervalo máximo de agregação de um minuto

O `create-flow-logs` exemplo a seguir cria um log de fluxo que captura todo o tráfego para o especificado VPC e entrega os logs de fluxo para um bucket do Amazon S3. O `--max-aggregation-interval` parâmetro especifica um intervalo máximo de agregação de 60 segundos (1 minuto).

```
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
```

Para obter mais informações, consulte [Logs de VPC fluxo](#) no Guia VPC do usuário da Amazon.

- Para API obter detalhes, consulte [CreateFlowLogs](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo cria um EC2 registro de fluxo da sub-rede subnet-1d234567 até o cloud-watch-log nome 'subnet1-log' para todo o tráfego " usando as permissões da função 'Admin' REJECT

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

Saída:

```
ClientToken                               FlowLogIds                               Unsuccessful
-----
m1VN2cxP3iB4qo//VUK15EU6cF7gQL0xcqNefvjeTGw= {f1-012fc34eed5678c9d} {}
```

- Para API obter detalhes, consulte [CreateFlowLogsem](#) Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **CreateImage** com um CLI

Os exemplos de código a seguir mostram como usar o CreateImage.

CLI

AWS CLI

Exemplo 1: Para criar um a AMI partir de uma instância EBS apoiada pela Amazon

O create-image exemplo a seguir cria um AMI a partir da instância especificada.

```
aws ec2 create-image \
```

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

Saída:

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

Para obter mais informações sobre como especificar um mapeamento de dispositivos de blocos para vocêAMI, consulte [Especificação de um mapeamento de dispositivos de blocos para um AMI no Guia EC2](#) do usuário da Amazon.

Exemplo 2: Para criar um a AMI partir de uma instância EBS apoiada pela Amazon sem reinicializar

O `create-image` exemplo a seguir cria AMI e define o parâmetro `--no-reboot`, para que a instância não seja reinicializada antes da criação da imagem.

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

Saída:

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

Para obter mais informações sobre como especificar um mapeamento de dispositivos de blocos para vocêAMI, consulte [Especificação de um mapeamento de dispositivos de blocos para um AMI no Guia EC2](#) do usuário da Amazon.

Exemplo 3: Para marcar um AMI e tirar fotos na criação

O `create-image` exemplo a seguir cria umAMI, e marca o AMI e os instantâneos com a mesma tag `cost-center=cc123`

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

```
--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}]"
```

Saída:

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

Para obter mais informações sobre como marcar seus recursos na criação, consulte [Adicionar tags na criação de recursos](#) no Guia do EC2 usuário da Amazon.

- Para API obter detalhes, consulte [CreateImagem](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo cria um AMI com o nome e a descrição especificados, a partir da instância especificada. A Amazon EC2 tenta desligar completamente a instância antes de criar a imagem e reinicia a instância após a conclusão.

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

Exemplo 2: Este exemplo cria um AMI com o nome e a descrição especificados, a partir da instância especificada. EC2A Amazon cria a imagem sem desligar e reiniciar a instância; portanto, a integridade do sistema de arquivos na imagem criada não pode ser garantida.

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

Exemplo 3: Este exemplo cria um AMI com três volumes. O primeiro volume é baseado em um EBS snapshot da Amazon. O segundo volume é um volume vazio de 100 GiB da AmazonEBS. O terceiro volume é um volume de armazenamento de instâncias. A sintaxe usada neste exemplo requer a PowerShell versão 3 ou superior.


```
$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"})
```

- Para API obter detalhes, consulte [CreateImage](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use `CreateInstanceExportTask` com um CLI

Os exemplos de código a seguir mostram como usar o `CreateInstanceExportTask`.

CLI

AWS CLI

Para exportar uma instância

Esse exemplo de comando cria uma tarefa para exportar a instância `i-1234567890abcdef0` para o bucket `myexportbucket` do Amazon S3.

Comando:

```
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
```

Saída:

```
{
  "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"
}
}

```

- Para API obter detalhes, consulte [CreateInstanceExportTask](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo exporta uma instância parada, **i-0800b00a00EXAMPLE**, como um disco rígido virtual (VHD) para o bucket **testbucket-export-instances-2019** do S3. O ambiente de destino é **Microsoft**, e o parâmetro `region` é adicionado porque a instância está na **us-east-1** região, enquanto a AWS região padrão do usuário não é `us-east-1`. Para obter o status da tarefa de exportação, copie o **ExportTaskId** valor dos resultados desse comando e execute **Get-EC2ExportTask -ExportTaskId export_task_ID_from_results**.

```

New-EC2InstanceExportTask -InstanceId i-0800b00a00EXAMPLE -
ExportToS3Task_DiskImageFormat VHD -ExportToS3Task_S3Bucket "amzn-s3-demo-bucket"
-TargetEnvironment Microsoft -Region us-east-1

```

Saída:

```

Description           :
ExportTaskId          : export-i-077c73108aEXAMPLE
ExportToS3Task        : Amazon.EC2.Model.ExportToS3Task
InstanceExportDetails : Amazon.EC2.Model.InstanceExportDetails
State                  : active
StatusMessage         :

```

- Para API obter detalhes, consulte [CreateInstanceExportTask](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use `CreateInternetGateway` com um CLI

Os exemplos de código a seguir mostram como usar o `CreateInternetGateway`.

CLI

AWS CLI

Para criar um gateway de internet

O `create-internet-gateway` exemplo a seguir cria um gateway de internet com a `tagName=my-igw`.

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

Saída:

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

Para obter mais informações, consulte [Gateways de Internet](#) no Guia do VPC usuário da Amazon.

- Para API obter detalhes, consulte [CreateInternetGateway](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo cria um gateway de Internet.

```
New-EC2InternetGateway
```

Saída:

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

- Para API obter detalhes, consulte [CreateInternetGateway](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **CreateKeyPair** com um AWS SDK ou CLI

Os exemplos de código a seguir mostram como usar o `CreateKeyPair`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação no contexto no seguinte exemplo de código:

- [Aprender os conceitos básicos](#)

.NET

AWS SDK for .NET

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
/// <summary>
/// Create an Amazon EC2 key pair with a specified name.
/// </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)
{
    try
    {
        var request = new CreateKeyPairRequest { KeyName = keyPairName, };

        var response = await _amazonEC2.CreateKeyPairAsync(request);

        var kp = response.KeyPair;
        // Return the key pair so it can be saved if needed.

        // Wait until the key pair exists.
        int retries = 5;
        while (retries-- > 0)
        {
            Console.WriteLine($"Checking for new KeyPair {keyPairName}...");
            var keyPairs = await DescribeKeyPairs(keyPairName);
            if (keyPairs.Any())
            {
                return kp;
            }

            Thread.Sleep(5000);
            retries--;
        }
        _logger.LogError($"Unable to find newly created KeyPair
{keyPairName}.");
    }
}
```

```
        throw new DoesNotExistException("KeyPair not found");
    }
    catch (AmazonEC2Exception ec2Exception)
    {
        if (ec2Exception.ErrorCode == "InvalidKeyPair.Duplicate")
        {
            _logger.LogError(
                $"A key pair called {keyPairName} already exists.");
        }

        throw;
    }
    catch (Exception ex)
    {
        _logger.LogError(
            $"An error occurred while creating the key pair.: {ex.Message}");
        throw;
    }
}

/// <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;
}
```

- Para API obter detalhes, consulte [CreateKeyPair](#) em AWS SDK for .NET API Referência.

Bash

AWS CLI com script Bash

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
#####
# 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 getopt 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 getopt "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
}
```


As funções utilitárias usadas neste exemplo.

```
#####
# 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
}
```

- Para API obter detalhes, consulte [CreateKeyPair](#) na Referência de AWS CLI Comandos.

C++

SDK para C++

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
#!/ Create an Amazon Elastic Compute Cloud (Amazon EC2) instance key pair.
/*!
 \param keyPairName: A name for a key pair.
 \param keyFilePath: File path where the credentials are stored. Ignored if it
 is an empty string;
 \param clientConfiguration: AWS client configuration.
 \return bool: Function succeeded.
 */
bool AwsDoc::EC2::createKeyPair(const Aws::String &keyPairName, const Aws::String
&keyFilePath,
                               const Aws::Client::ClientConfiguration
&clientConfiguration) {
    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 - " << keyPairName << ". " <<
outcome.GetError().GetMessage() << std::endl;
    } else {
        std::cout << "Successfully created key pair named " <<
keyPairName << std::endl;
        if (!keyFilePath.empty()) {
```

```
        std::ofstream keyFile(keyFilePath.c_str());
        keyFile << outcome.GetResult().GetKeyMaterial();
        keyFile.close();
        std::cout << "Keys written to the file " <<
                    keyFilePath << std::endl;
    }

}

return outcome.IsSuccess();

}
```

- Para API obter detalhes, consulte [CreateKeyPair](#) em AWS SDK for C++ API Referência.

CLI

AWS CLI

Para criar um par de chaves

Este exemplo cria um par de chaves denominado `MyKeyPair`.

Comando:

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


A saída é uma ASCII versão da chave privada e da impressão digital da chave. Você precisa salvar a chave em um arquivo.

Para obter mais informações, consulte [Using Key Pairs](#) no Guia do usuário da AWS Command Line Interface.

- Para API obter detalhes, consulte [CreateKeyPair](#) na Referência de AWS CLI Comandos.

Java

SDK para Java 2.x

 Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
/**
 * Creates a new key pair asynchronously.
 *
 * @param keyName the name of the key pair to create
 * @param fileName the name of the file to write the key material to
 * @return a {@link CompletableFuture} that represents the asynchronous
operation
 *         of creating the key pair and writing the key material to a file
 */
public CompletableFuture<CreateKeyPairResponse> createKeyPairAsync(String
keyName, String fileName) {
    CreateKeyPairRequest request = CreateKeyPairRequest.builder()
        .keyName(keyName)
        .build();

    CompletableFuture<CreateKeyPairResponse> responseFuture =
getAsyncClient().createKeyPair(request);
    responseFuture.whenComplete((response, exception) -> {
        if (response != null) {
            try {
                BufferedWriter writer = new BufferedWriter(new
FileWriter(fileName));
                writer.write(response.keyMaterial());
                writer.close();
            } catch (IOException e) {
                throw new RuntimeException("Failed to write key material to
file: " + e.getMessage(), e);
            }
        } else {
            throw new RuntimeException("Failed to create key pair: " +
exception.getMessage(), exception);
        }
    });
}
```

```
    });  
  
    return responseFuture;  
}
```

- Para API obter detalhes, consulte [CreateKeyPair](#) em AWS SDK for Java 2.x API Referência.

JavaScript

SDK para JavaScript (v3)

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
import { CreateKeyPairCommand, EC2Client } from "@aws-sdk/client-ec2";  
  
/**  
 * Creates an ED25519 or 2048-bit RSA key pair with the specified name and in the  
 * specified PEM or PPK format.  
 * Amazon EC2 stores the public key and displays the private key for you to save  
 * to a file.  
 * @param {{ keyName: string }} options  
 */  
export const main = async ({ keyName }) => {  
  const client = new EC2Client({});  
  const command = new CreateKeyPairCommand({  
    KeyName: keyName,  
  });  
  
  try {  
    const { KeyMaterial, KeyName } = await client.send(command);  
    console.log(KeyName);  
    console.log(KeyMaterial);  
  } catch (caught) {  
    if (caught instanceof Error && caught.name === "InvalidKeyPair.Duplicate") {  
      console.warn(`${caught.message}. Try another key name.`);  
    } else {  
      throw caught;  
    }  
  }  
}
```

```
    }  
  }  
};
```

- Para API obter detalhes, consulte [CreateKeyPair](#) em AWS SDK for JavaScript API Referência.

Kotlin

SDK para Kotlin

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
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}")  
    }  
}
```

- Para API obter detalhes, consulte a [CreateKeyPair](#) referência AWS SDK do Kotlin API.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo cria um par de chaves e captura a chave RSA privada PEM codificada em -em um arquivo com o nome especificado. Quando você estiver usando PowerShell, a codificação deve ser definida como ascii para gerar uma chave válida. Para

obter mais informações, consulte Criar, exibir e excluir pares de EC2 chaves da Amazon (<https://docs.aws.amazon.com/cli/latest/userguide/cli-services-ec2-keypairs.html>) no Guia do usuário da interface de linha de AWS comando.

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

- Para API obter detalhes, consulte [CreateKeyPair](#) em Referência de AWS Tools for PowerShell cmdlet.

Python

SDK para Python (Boto3)

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
class KeyPairWrapper:
    """
    Encapsulates Amazon Elastic Compute Cloud (Amazon EC2) key pair actions.
    This class provides methods to create, list, and delete EC2 key pairs.
    """

    def __init__(
        self,
        ec2_client: boto3.client,
        key_file_dir: Union[tempfile.TemporaryDirectory, str],
        key_pair: Optional[dict] = None,
    ):
        """
        Initializes the KeyPairWrapper with the specified EC2 client, key file
        directory,
        and an optional key pair.

        :param ec2_client: A Boto3 Amazon EC2 client. This client provides low-
        level
                           access to AWS EC2 services.
```

```

        :param key_file_dir: The folder where the private key information is
        stored.
            This should be a secure folder.
        :param key_pair: A dictionary representing the Boto3 KeyPair object.
            This is a high-level object that wraps key pair actions.
Optional.
    """
    self.ec2_client = ec2_client
    self.key_pair = key_pair
    self.key_file_path: Optional[str] = None
    self.key_file_dir = key_file_dir

    @classmethod
    def from_client(cls) -> "KeyPairWrapper":
        """
        Class method to create an instance of KeyPairWrapper using a new EC2
client
and a temporary directory for storing key files.

        :return: An instance of KeyPairWrapper.
        """
        ec2_client = boto3.client("ec2")
        return cls(ec2_client, tempfile.TemporaryDirectory())

    def create(self, key_name: str) -> dict:
        """
        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 dictionary representing the Boto3 KeyPair object that
represents the newly created key pair.
        :raises ClientError: If there is an error in creating the key pair, for
example, if a key pair with the same name already exists.
        """
        try:
            response = self.ec2_client.create_key_pair(KeyName=key_name)
            self.key_pair = response
            self.key_file_path = os.path.join(
                self.key_file_dir.name, f"{self.key_pair['KeyName']}.pem"

```



```
    )
    with open(self.key_file_path, "w") as key_file:
        key_file.write(self.key_pair["KeyMaterial"])
except ClientError as err:
    if err.response["Error"]["Code"] == "InvalidKeyPair.Duplicate":
        logger.error(
            f"A key pair called {key_name} already exists. "
            "Please choose a different name for your key pair "
            "or delete the existing key pair before creating."
        )
    raise
else:
    return self.key_pair
```

- Para API obter detalhes, consulte a [CreateKeyPair](#) Referência AWS SDK do Python (Boto3).
API

Ruby

SDK para Ruby

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
# 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}'."
  true
rescue Aws::EC2::Errors::InvalidKeyPairDuplicate
  puts "Error creating key pair: a key pair named '#{key_pair_name}' " \
    'already exists.'
  false
rescue StandardError => e
  puts "Error creating key pair or saving private key file: #{e.message}"
  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
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)
  true
rescue StandardError => e
  puts "Error deleting key pair: #{e.message}"
  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__
```

- Para API obter detalhes, consulte [CreateKeyPair](#) em AWS SDK for Ruby API Referência.

Rust

SDK para Rust

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

Implementação do Rust que chama o `create_key_pair` do EC2 cliente e extrai o material retornado.

```
pub async fn create_key_pair(&self, name: String) -> Result<(KeyPairInfo,
String), EC2Error> {
    tracing::info!("Creating key pair {name}");
    let output = self.client.create_key_pair().key_name(name).send().await?;
    let info = KeyPairInfo::builder()
        .set_key_name(output.key_name)
        .set_key_fingerprint(output.key_fingerprint)
        .set_key_pair_id(output.key_pair_id)
        .build();
    let material = output
        .key_material
        .ok_or_else(|| EC2Error::new("Create Key Pair has no key
material"))?;
    Ok((info, material))
}
```

Uma função que chama o `create_key` impl e salva com segurança a chave privada. PEM

```
/// 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.
pub async fn create(
    &mut self,
    ec2: &EC2,
```

```

        util: &Util,
        key_name: String,
    ) -> Result<KeyPairInfo, EC2Error> {
        let (key_pair, material) = ec2
            .create_key_pair(key_name.clone())
            .await
            .map_err(|e| e.add_message(format!("Couldn't create key
{key_name}")))?;

        let path = self.key_file_dir.join(format!("{key_name}.pem"));

        util.write_secure(&key_name, &path, material)?;

        self.key_file_path = Some(path);
        self.key_pair = key_pair.clone();

        Ok(key_pair)
    }

```

- Para API obter detalhes, consulte [CreateKeyPair](#) AWS SDKAPIreferência do Rust.

SAP ABAP

SDKpara SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```

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.

```

- Para API obter detalhes, consulte [CreateKeyPair AWS SDK](#) para SAP ABAP API referência.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **CreateLaunchTemplate** com um AWS SDK ou CLI

Os exemplos de código a seguir mostram como usar o `CreateLaunchTemplate`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação no contexto no seguinte exemplo de código:

- [Criar e gerenciar um serviço resiliente](#)

.NET

AWS SDK for .NET

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
/// <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)
{
    try
    {
        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;
    }
    catch (AmazonEC2Exception ec2Exception)
    {
        if (ec2Exception.ErrorCode ==
"InvalidLaunchTemplateName.AlreadyExistsException")
```



```

        {
            _logger.LogError($"Could not create the template, the name
{_launchTemplateName} already exists. " +
                $"Please try again with a unique name.");
        }

        throw;
    }
    catch (Exception ex)
    {
        _logger.LogError($"An error occurred while creating the template.:
{ex.Message}");
        throw;
    }
}

```

- Para API obter detalhes, consulte [CreateLaunchTemplate](#) em AWS SDK for .NET API Referência.

CLI

AWS CLI

Exemplo 1: criar um modelo de lançamento

O `create-launch-template` exemplo a seguir cria um modelo de execução que especifica a sub-rede na qual iniciar a instância, atribui um endereço IP público e um IPv6 endereço à instância e cria uma tag para a instância.

```

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"}]}]}'

```

Saída:

```

{
  "LaunchTemplate": {
    "LatestVersionNumber": 1,

```

```

    "LaunchTemplateId": "lt-01238c059e3466abc",
    "LaunchTemplateName": "TemplateForWebServer",
    "DefaultVersionNumber": 1,
    "CreatedBy": "arn:aws:iam::123456789012:user/Bob",
    "CreateTime": "2019-01-27T09:13:24.000Z"
  }
}

```

Para obter mais informações, consulte Execução de uma instância em um modelo de execução no Guia do usuário do Amazon Elastic Compute Cloud. Para obter informações sobre como citar parâmetros JSON formatados, consulte Citar cadeias de caracteres no Guia do usuário da interface de linha de AWS comando.

Exemplo 2: Para criar um modelo de lançamento para o Amazon EC2 Auto Scaling

O `create-launch-template` exemplo a seguir cria um modelo de execução com várias tags e um mapeamento de dispositivos de blocos para especificar um EBS volume adicional quando uma instância é executada. Especifique um valor `Groups` que corresponda aos grupos de segurança nos VPC quais seu grupo de Auto Scaling iniciará instâncias. Especifique as sub-redes VPC e como propriedades do grupo Auto Scaling.

```

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

```

Saída:

```

{
  "LaunchTemplate": {
    "LatestVersionNumber": 1,
    "LaunchTemplateId": "lt-0123c79c33a54e0abc",
    "LaunchTemplateName": "TemplateForAutoScaling",
    "DefaultVersionNumber": 1,

```

```
    "CreatedBy": "arn:aws:iam::123456789012:user/Bob",
    "CreateTime": "2019-04-30T18:16:06.000Z"
  }
}
```

Para obter mais informações, consulte Criação de um modelo de lançamento para um grupo de Auto Scaling no Guia do usuário do Amazon Auto EC2 Scaling. Para obter informações sobre como citar parâmetros JSON formatados, consulte Citar cadeias de caracteres no Guia do usuário da interface de linha de AWS comando.

Exemplo 3: Para criar um modelo de lançamento que especifique a criptografia de volumes EBS

O `create-launch-template` exemplo a seguir cria um modelo de execução que inclui EBS volumes criptografados criados a partir de um snapshot não criptografado. Ele também coloca tags nos volumes durante a criação. Se a criptografia por padrão estiver desabilitada, você deve especificar a opção `"Encrypted"` conforme mostrado no exemplo a seguir. Se você usar a `"KmsKeyId"` opção para especificar um cliente gerenciado CMK, também deverá especificar a `"Encrypted"` opção, mesmo que a criptografia esteja ativada por padrão.

```
aws ec2 create-launch-template \  
  --launch-template-name TemplateForEncryption \  
  --launch-template-data file://config.json
```

Conteúdo de `config.json`:

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

Saída:

```
{
  "LaunchTemplate": {
    "LatestVersionNumber": 1,
    "LaunchTemplateId": "lt-0d5bd51bcf8530abc",
    "LaunchTemplateName": "TemplateForEncryption",
    "DefaultVersionNumber": 1,
    "CreatedBy": "arn:aws:iam::123456789012:user/Bob",
    "CreateTime": "2020-01-07T19:08:36.000Z"
  }
}
```

Para obter mais informações, consulte [Restauração de um EBS volume da Amazon a partir de um snapshot e criptografia por padrão](#) no Guia do usuário do Amazon Elastic Compute Cloud.

- Para API obter detalhes, consulte [CreateLaunchTemplate](#) na Referência de AWS CLI Comandos.

JavaScript

SDK para JavaScript (v3)

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
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,
    },
  }),
);
```

- Para API obter detalhes, consulte [CreateLaunchTemplate](#) em AWS SDK for JavaScript API Referência.

Python

SDK para Python (Boto3)

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

Este exemplo cria um modelo de execução que inclui um perfil de instância que concede permissões específicas à instância e um script Bash de dados do usuário que é executado na instância após sua inicialização.

```
class AutoScalingWrapper:
    """
```

```
Encapsulates Amazon EC2 Auto Scaling and EC2 management actions.
"""

def __init__(
    self,
    resource_prefix: str,
    inst_type: str,
    ami_param: str,
    autoscaling_client: boto3.client,
    ec2_client: boto3.client,
    ssm_client: boto3.client,
    iam_client: boto3.client,
):
    """
    Initializes the AutoScaler class with the necessary parameters.

    :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
    sts_client = boto3.client("sts")
    self.account_id = sts_client.get_caller_identity()["Account"]

    self.key_pair_name = f"{resource_prefix}-key-pair"
    self.launch_template_name = f"{resource_prefix}-template-"
    self.group_name = f"{resource_prefix}-group"

    # Happy path
    self.instance_policy_name = f"{resource_prefix}-pol"
    self.instance_role_name = f"{resource_prefix}-role"
    self.instance_profile_name = f"{resource_prefix}-prof"
```

```
# Failure mode
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"

def create_template(
    self, server_startup_script_file: str, instance_policy_file: str
) -> Dict[str, Any]:
    """
    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:
        # Create key pair and instance profile
        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,
        )

        # Read the startup script
        with open(server_startup_script_file) as file:
            start_server_script = file.read()

        # Get the latest AMI ID
        ami_latest = self.ssm_client.get_parameter(Name=self.ami_param)
```

```
ami_id = ami_latest["Parameter"]["Value"]

# Create the launch template
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(
    f"Created launch template {self.launch_template_name} for AMI
{ami_id} on {self.inst_type}."
)
except ClientError as err:
    log.error(f"Failed to create launch template
{self.launch_template_name}.")
    error_code = err.response["Error"]["Code"]
    if error_code == "InvalidLaunchTemplateName.AlreadyExistsException":
        log.info(
            f"Launch template {self.launch_template_name} already exists,
nothing to do."
        )
    log.error(f"Full error:\n\t{err}")
return template
```

- Para API obter detalhes, consulte a [CreateLaunchTemplate](#) Referência AWS SDK do Python (Boto3). API

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use `CreateNetworkAcl` com um CLI

Os exemplos de código a seguir mostram como usar o `CreateNetworkAcl`.

CLI

AWS CLI

Para criar uma rede ACL

Este exemplo cria uma rede ACL para o especificadoVPC.

Comando:

```
aws ec2 create-network-acl --vpc-id vpc-a01106c2
```

Saída:

```
{
  "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
  }
}
```

- Para API obter detalhes, consulte [CreateNetworkAcl](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo cria uma rede ACL para o especificadoVPC.

```
New-EC2NetworkAcl -VpcId vpc-12345678
```

Saída:

```
Associations : {}
Entries      : {Amazon.EC2.Model.NetworkAclEntry,
               Amazon.EC2.Model.NetworkAclEntry}
IsDefault   : False
NetworkAclId : acl-12345678
Tags        : {}
VpcId       : vpc-12345678
```

- Para API obter detalhes, consulte [CreateNetworkAcl](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **CreateNetworkAclEntry** com um CLI

Os exemplos de código a seguir mostram como usar o `CreateNetworkAclEntry`.

CLI

AWS CLI

Para criar uma ACL entrada de rede

Este exemplo cria uma entrada para a rede especificadaACL. A regra permite o tráfego de entrada de qualquer IPv4 endereço (0.0.0.0/0) na UDP porta 53 (DNS) em qualquer sub-rede associada. Se o comando for bem-sucedido, nenhuma saída será retornada.

Comando:

```
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
```

Este exemplo cria uma regra para a rede especificada ACL que permite o tráfego de entrada de qualquer IPv6 endereço (:: /0) na TCP porta 80 (HTTP).

Comando:

```
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
```

- Para API obter detalhes, consulte [CreateNetworkAclEntry](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo cria uma entrada para a rede especificada ACL. A regra permite tráfego de entrada de qualquer lugar (0.0.0.0/0) na UDP porta 53 (DNS) em qualquer sub-rede associada.

```
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
```

- Para API obter detalhes, consulte [CreateNetworkAclEntry](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **CreateNetworkInterface** com um CLI

Os exemplos de código a seguir mostram como usar o `CreateNetworkInterface`.

CLI

AWS CLI

Exemplo 1: Para especificar um IPv4 endereço para uma interface de rede

O `create-network-interface` exemplo a seguir cria uma interface de rede para a sub-rede especificada com o IPv4 endereço primário especificado.

```
aws ec2 create-network-interface \  
  --subnet-id subnet-00a24d0d67acf6333 \  
  --description "my network interface" \  
  --groups sg-09dfba7ed20cda78b \  
  --private-ip-address 10.0.8.17
```

Saída:

```
{  
  "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"
  }
}

```

Exemplo 2: Para criar uma interface de rede com um IPv4 endereço e um IPv6 endereço

O `create-network-interface` exemplo a seguir cria uma interface de rede para a sub-rede especificada com um IPv4 endereço e um IPv6 endereço selecionados pela AmazonEC2.

```

aws ec2 create-network-interface \
  --subnet-id subnet-00a24d0d67acf6333 \
  --description "my dual stack network interface" \
  --ipv6-address-count 1 \
  --groups sg-09dfba7ed20cda78b

```

Saída:

```

{
  "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-02723a0feeb9d57b",
    "Ipv6Address": "2600:1f13:cfe:3650:a1dc:237c:393a:4ba7"
  }
}

```

Exemplo 3: Para criar uma interface de rede com opções de configuração de rastreamento de conexão

O `create-network-interface` exemplo a seguir cria uma interface de rede e configura os tempos limite de rastreamento de conexão ociosa.

```

aws ec2 create-network-interface \
  --subnet-id subnet-00a24d0d67acf6333 \
  --groups sg-02e57dbcfe0331c1b \
  --connection-tracking-specification TcpEstablishedTimeout=86400,UdpTimeout=60

```

Saída:

```

{
  "NetworkInterface": {
    "AvailabilityZone": "us-west-2a",
    "ConnectionTrackingConfiguration": {
      "TcpEstablishedTimeout": 86400,
      "UdpTimeout": 60
    },
    "Description": "",
    "Groups": [

```

```

    {
      "GroupName": "my-security-group",
      "GroupId": "sg-02e57dbcf0331c1b"
    }
  ],
  "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-02723a0feeb9d57b"
}
}

```

Exemplo 4: Para criar um adaptador de malha elástica

O `create-network-interface` exemplo a seguir cria um EFA.

```

aws ec2 create-network-interface \
  --interface-type efa \
  --subnet-id subnet-00a24d0d67acf6333 \
  --description "my efa" \
  --groups sg-02e57dbcf0331c1b

```

Saída:

```

{
  "NetworkInterface": {

```

```
"AvailabilityZone": "us-west-2a",
"Description": "my efa",
"Groups": [
  {
    "GroupName": "my-efa-sg",
    "GroupId": "sg-02e57dbcf0331c1b"
  }
],
"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-02723a0feeb9d57b"
}
}
```

Para obter mais informações, consulte [Interfaces de rede elásticas](#) no Guia EC2 do usuário da Amazon.

- Para API obter detalhes, consulte [CreateNetworkInterface](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo cria a interface de rede especificada.


```
New-EC2NetworkInterface -SubnetId subnet-1a2b3c4d -Description "my network
interface" -Group sg-12345678 -PrivateIpAddress 10.0.0.17
```

Saída:

```
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
```

- Para API obter detalhes, consulte [CreateNetworkInterface](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **CreatePlacementGroup** com um CLI

Os exemplos de código a seguir mostram como usar o `CreatePlacementGroup`.

CLI

AWS CLI

Para criar um grupo de colocação

Esse exemplo de comando cria um grupo de posicionamento com o nome especificado.

Comando:

```
aws ec2 create-placement-group --group-name my-cluster --strategy cluster
```

Para criar um grupo de posicionamento de partições

Esse exemplo de comando cria um grupo de posicionamento de partições chamado HDFS-Group-A com cinco partições.

Comando:

```
aws ec2 create-placement-group --group-name HDFS-Group-A --strategy partition --partition-count 5
```

- Para API obter detalhes, consulte [CreatePlacementGroup](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Esse exemplo cria um grupo de posicionamento com o nome especificado.

```
New-EC2PlacementGroup -GroupName my-placement-group -Strategy cluster
```

- Para API obter detalhes, consulte [CreatePlacementGroup](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **CreateRoute** com um CLI

Os exemplos de código a seguir mostram como usar o CreateRoute.

CLI

AWS CLI

Para criar uma rota

Este exemplo cria uma rota para a tabela de rotas especificada. A rota corresponde a todo o IPv4 tráfego ($0.0.0.0/0$) e o encaminha para o gateway de Internet especificado. Se o comando for bem-sucedido, nenhuma saída será retornada.

Comando:

```
aws ec2 create-route --route-table-id rtb-22574640 --destination-cidr-block 0.0.0.0/0 --gateway-id igw-c0a643a9
```

Este exemplo de comando cria uma rota na tabela de rotas *rtb-g8ff4ea2*. A rota corresponde ao tráfego do IPv4 CIDR bloco $10.0.0.0/16$ e o encaminha para a conexão de VPC peering, *pcx-111aaa22*. Essa rota permite que o tráfego seja direcionado para o par VPC na conexão de VPC peering. Se o comando for bem-sucedido, nenhuma saída será retornada.

Comando:

```
aws ec2 create-route --route-table-id rtb-g8ff4ea2 --destination-cidr-block 10.0.0.0/16 --vpc-peering-connection-id pcx-1a2b3c4d
```

Este exemplo cria uma rota na tabela de rotas especificada que corresponde a todo o IPv6 tráfego ($:::/0$) e a encaminha para o gateway de Internet somente de saída especificado.

Comando:

```
aws ec2 create-route --route-table-id rtb-dce620b8 --destination-ipv6-cidr-block :::/0 --egress-only-internet-gateway-id eigw-01eadbd45ecd7943f
```

- Para API obter detalhes, consulte [CreateRoute](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo cria a rota especificada para a tabela de rotas especificada. A rota corresponde a todo o tráfego e o envia para o gateway de Internet especificado.

```
New-EC2Route -RouteTableId rtb-1a2b3c4d -DestinationCidrBlock 0.0.0.0/0 -GatewayId igw-1a2b3c4d
```

Saída:

```
True
```

- Para API obter detalhes, consulte [CreateRoute](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use `CreateRouteTable` com um AWS SDK ou CLI

Os exemplos de código a seguir mostram como usar o `CreateRouteTable`.

CLI

AWS CLI

Para criar uma tabela de rotas

Este exemplo cria uma tabela de rotas para o especificado VPC.

Comando:

```
aws ec2 create-route-table --vpc-id vpc-a01106c2
```

Saída:

```
{
  "RouteTable": {
    "Associations": [],
    "RouteTableId": "rtb-22574640",
    "VpcId": "vpc-a01106c2",
    "PropagatingVgws": [],
    "Tags": [],
    "Routes": [
      {
        "GatewayId": "local",
        "DestinationCidrBlock": "10.0.0.0/16",
        "State": "active"
      }
    ]
  }
}
```

```
}
```

- Para API obter detalhes, consulte [CreateRouteTable](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo cria uma tabela de rotas para o especificado VPC.

```
New-EC2RouteTable -VpcId vpc-12345678
```

Saída:

```
Associations      : {}  
PropagatingVgws  : {}  
Routes           : {}  
RouteTableId     : rtb-1a2b3c4d  
Tags             : {}  
VpcId            : vpc-12345678
```

- Para API obter detalhes, consulte [CreateRouteTable](#) em Referência de AWS Tools for PowerShell cmdlet.

Ruby

SDK para Ruby

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
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}'."
  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.'
  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__
```

- Para API obter detalhes, consulte [CreateRouteTable](#) em AWS SDK for Ruby API Referência.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **CreateSecurityGroup** com um AWS SDK ou CLI


Os exemplos de código a seguir mostram como usar o CreateSecurityGroup.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação no contexto no seguinte exemplo de código:

- [Aprender os conceitos básicos](#)

.NET

AWS SDK for .NET

 Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
/// <summary>
/// Create an Amazon EC2 security group with a specified name and
description.
/// </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)
{
    try
    {
        var response = await _amazonEC2.CreateSecurityGroupAsync(
            new CreateSecurityGroupRequest(groupName, groupDescription));

        // Wait until the security group exists.
        int retries = 5;
        while (retries-- > 0)
        {
            var groups = await DescribeSecurityGroups(response.GroupId);
            if (groups.Any())
            {
                return response.GroupId;
            }
        }
    }
}
```

```

        Thread.Sleep(5000);
        retries--;
    }
    _logger.LogError($"Unable to find newly created group {groupName}.");
    throw new DoesNotExistException("security group not found");
}
catch (AmazonEC2Exception ec2Exception)
{
    if (ec2Exception.ErrorCode == "ResourceAlreadyExists")
    {
        _logger.LogError(
            $"A security group with the name {groupName} already exists.
{ec2Exception.Message}");
    }
    throw;
}
catch (Exception ex)
{
    _logger.LogError(
        $"An error occurred while creating the security group.:
{ex.Message}");
    throw;
}
}
}

```

- Para API obter detalhes, consulte [CreateSecurityGroup](#) em AWS SDK for .NET APIReferência.

Bash

AWS CLI com script Bash

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
#####
```

```

# 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
}

```

```

    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
}

```

As funções utilitárias usadas neste exemplo.

```

#####
# 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
}
```

- Para API obter detalhes, consulte [CreateSecurityGroup](#) na Referência de AWS CLI Comandos.

C++

SDK para C++

 Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
#!/ Create a security group.
/*!
  \param groupName: A security group name.
  \param description: A description.
  \param vpcID: A virtual private cloud (VPC) ID.
  \param[out] groupIDResult: A string to receive the group ID.
  \param clientConfiguration: AWS client configuration.
  \return bool: Function succeeded.
*/
bool AwsDoc::EC2::createSecurityGroup(const Aws::String &groupName,
                                     const Aws::String &description,
                                     const Aws::String &vpcID,
                                     Aws::String &groupIDResult,
                                     const Aws::Client::ClientConfiguration
&clientConfiguration) {
    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;

groupIDResult = outcome.GetResult().GetGroupId();

return true;
}
```

- Para API obter detalhes, consulte [CreateSecurityGroup](#) em AWS SDK for C++ API Referência.

CLI

AWS CLI

Para criar um grupo de segurança para EC2 -Classic

Este exemplo cria um grupo de segurança chamado MySecurityGroup.

Comando:

```
aws ec2 create-security-group --group-name MySecurityGroup --description "My security group"
```

Saída:

```
{
  "GroupId": "sg-903004f8"
}
```

Para criar um grupo de segurança para EC2 - VPC

Este exemplo cria um grupo de segurança com MySecurityGroup o nome especificadoVPC.

Comando:

```
aws ec2 create-security-group --group-name MySecurityGroup --description "My security group" --vpc-id vpc-1a2b3c4d
```

Saída:

```
{
  "GroupId": "sg-903004f8"
}
```

Para obter mais informações, consulte [Using Security Groups](#) no Guia do usuário da AWS Command Line Interface.

- Para API obter detalhes, consulte [CreateSecurityGroup](#) na Referência de AWS CLI Comandos.

Java**SDK para Java 2.x****Note**

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
/**
 * Creates a new security group asynchronously with the specified group name,
 * description, and VPC ID. It also
 *   * authorizes inbound traffic on ports 80 and 22 from the specified IP
 * address.
 *
 * @param groupName the name of the security group to create
 * @param groupDesc the description of the security group
 * @param vpcId the ID of the VPC in which to create the security
 * group
 * @param myIpAddress the IP address from which to allow inbound traffic
 * (e.g., "192.168.1.1/0" to allow traffic from
 *   * any IP address in the 192.168.1.0/24 subnet)
 * @return a CompletableFuture that, when completed, returns the ID of the
 * created security group
 * @throws RuntimeException if there was a failure creating the security
 * group or authorizing the inbound traffic
 */
```



```
public CompletableFuture<String> createSecurityGroupAsync(String groupName,
String groupDesc, String vpcId, String myIpAddress) {
    CreateSecurityGroupRequest createRequest =
CreateSecurityGroupRequest.builder()
        .groupName(groupName)
        .description(groupDesc)
        .vpcId(vpcId)
        .build();

    return getAsyncClient().createSecurityGroup(createRequest)
        .thenCompose(createResponse -> {
            String groupId = createResponse.groupId();
            IpRange ipRange = IpRange.builder()
                .cidrIp(myIpAddress + "/32")
                .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();

            return
getAsyncClient().authorizeSecurityGroupIngress(authRequest)
                .thenApply(authResponse -> groupId);
        })
        .whenComplete((result, exception) -> {
            if (exception != null) {
                if (exception instanceof CompletionException &&
exception.getCause() instanceof Ec2Exception) {
```

```
        throw (Ec2Exception) exception.getCause();
    } else {
        throw new RuntimeException("Failed to create security
group: " + exception.getMessage(), exception);
    }
    });
}
```

- Para API obter detalhes, consulte [CreateSecurityGroup](#) em AWS SDK for Java 2.x API Referência.

JavaScript

SDK para JavaScript (v3)

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
import { CreateSecurityGroupCommand, EC2Client } from "@aws-sdk/client-ec2";

/**
 * Creates a security group.
 * @param {{ groupName: string, description: string }} options
 */
export const main = async ({ groupName, description }) => {
    const client = new EC2Client({});
    const command = new CreateSecurityGroupCommand({
        // Up to 255 characters in length. Cannot start with sg-.
        GroupName: groupName,
        // Up to 255 characters in length.
        Description: description,
    });

    try {
        const { GroupId } = await client.send(command);
        console.log(GroupId);
    }
}
```

```
    } catch (caught) {
        if (caught instanceof Error && caught.name === "InvalidParameterValue") {
            console.warn(`${caught.message}.`);
        } else {
            throw caught;
        }
    }
};
```

- Para API obter detalhes, consulte [CreateSecurityGroup](#) em AWS SDK for JavaScript API Referência.

Kotlin

SDK para Kotlin

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
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
}
}
```

- Para API obter detalhes, consulte a [CreateSecurityGroup](#) referência AWS SDKdo Kotlin API.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo cria um grupo de segurança para o especificadoVPC.

```
New-EC2SecurityGroup -GroupName my-security-group -Description "my security
group" -VpcId vpc-12345678
```

Saída:

```
sg-12345678
```

Exemplo 2: Este exemplo cria um grupo de segurança para EC2 -Classic.

```
New-EC2SecurityGroup -GroupName my-security-group -Description "my security group"
```

Saída:

```
sg-45678901
```

- Para API obter detalhes, consulte [CreateSecurityGroup](#) em Referência de AWS Tools for PowerShell cmdlet.

Python

SDK para Python (Boto3)

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

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

    def __init__(self, ec2_client: boto3.client, security_group: Optional[str] =
    None):
        """
        Initializes the SecurityGroupWrapper with an EC2 client and an optional
        security group ID.

        :param ec2_client: A Boto3 Amazon EC2 client. This client provides low-
        level
                           access to AWS EC2 services.
        :param security_group: The ID of a security group to manage. This is a
        high-level identifier
                           that represents the security group.
```

```
    """
    self.ec2_client = ec2_client
    self.security_group = security_group

    @classmethod
    def from_client(cls) -> "SecurityGroupWrapper":
        """
        Creates a SecurityGroupWrapper instance with a default EC2 client.

        :return: An instance of SecurityGroupWrapper initialized with the default
        EC2 client.
        """
        ec2_client = boto3.client("ec2")
        return cls(ec2_client)

    def create(self, group_name: str, group_description: str) -> str:
        """
        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: The ID of the newly created security group.
        :raise Handles AWS SDK service-level ClientError, with special handling
        for ResourceAlreadyExists
        """
        try:
            response = self.ec2_client.create_security_group(
                GroupName=group_name, Description=group_description
            )
            self.security_group = response["GroupId"]
        except ClientError as err:
            if err.response["Error"]["Code"] == "ResourceAlreadyExists":
                logger.error(
                    f"Security group '{group_name}' already exists. Please choose
                    a different name."
                )
                raise
            else:
                return self.security_group
```

- Para API obter detalhes, consulte a [CreateSecurityGroup](#) Referência AWS SDK do Python (Boto3). API

Ruby

SDK para Ruby

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
# 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}'."
  security_group.group_id
rescue StandardError => e
  puts "Error creating security group: #{e.message}"
  '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}'."
  true
rescue StandardError => e
  puts "Error adding inbound rule to security group: #{e.message}"
  false
end

# Refactored method to simplify complexity for describing security group
permissions
def format_port_information(perm)
  from_port_str = perm.from_port == '-1' || perm.from_port == -1 ? 'All' :
perm.from_port.to_s
  to_port_str = perm.to_port == '-1' || perm.to_port == -1 ? 'All' :
perm.to_port.to_s
  { from_port: from_port_str, to_port: to_port_str }
end

# Displays information about a security group's IP permissions set in
# Amazon Elastic Compute Cloud (Amazon EC2).
def describe_security_group_permissions(perm)
  ports = format_port_information(perm)

  print " Protocol: #{perm.ip_protocol == '-1' ? 'All' : perm.ip_protocol}"
  print ", From: #{ports[:from_port]}, To: #{ports[:to_port]}"

  print ", CIDR IPv6: #{perm.ipv_6_ranges[0].cidr_ipv_6}" if perm.key?
(:ipv_6_ranges) && perm.ipv_6_ranges.count.positive?

```

```
print ", CIDR IPv4: #{perm.ip_ranges[0].cidr_ip}" if perm.key?(:ip_ranges) &&
perm.ip_ranges.count.positive?
print "\n"
end

# Displays information about available security groups in
# Amazon Elastic Compute Cloud (Amazon EC2).
def describe_security_groups(ec2_client)
  response = ec2_client.describe_security_groups

  if response.security_groups.count.positive?
    response.security_groups.each do |sg|
      display_group_details(sg)
    end
  else
    puts 'No security groups found.'
  end
rescue StandardError => e
  puts "Error getting information about security groups: #{e.message}"
end

# Helper method to display the details of security groups
def display_group_details(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}"

  display_group_tags(sg.tags) if sg.tags.count.positive?
  display_group_permissions(sg)
end

def display_group_tags(tags)
  puts 'Tags:'
  tags.each do |tag|
    puts "  Key: #{tag.key}, Value: #{tag.value}"
  end
end

def display_group_permissions(sg)
  if sg.ip_permissions.count.positive?
    puts 'Inbound rules:'
  end
end
```

```
    sg.ip_permissions.each do |p|
      describe_security_group_permissions(p)
    end
  end

  return if sg.ip_permissions_egress.empty?

  puts 'Outbound rules:'
  sg.ip_permissions_egress.each do |p|
    describe_security_group_permissions(p)
  end
end

# Deletes an Amazon Elastic Compute Cloud (Amazon EC2)
# security group.
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}'."
  true
rescue StandardError => e
  puts "Error deleting security group: #{e.message}"
  false
end

# Example usage with refactored run_me to reduce complexity
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 = process_arguments
  ec2_client = Aws::EC2::Client.new(region: region)

  security_group_id = attempt_create_security_group(ec2_client, group_name,
  description, vpc_id)
  security_group_exists = security_group_id != 'Error'

  if security_group_exists
    add_inbound_rules(ec2_client, security_group_id, ip_protocol_http,
  from_port_http, to_port_http, cidr_ip_range_http)
    add_inbound_rules(ec2_client, security_group_id, ip_protocol_ssh,
  from_port_ssh, to_port_ssh, cidr_ip_range_ssh)
  end

  describe_security_groups(ec2_client)
```

```
    attempt_delete_security_group(ec2_client, security_group_id) if
    security_group_exists
end

def process_arguments
  if ARGV[0] == '--help' || ARGV[0] == '-h'
    display_help
    exit 1
  elsif ARGV.count.zero?
    default_values
  else
    ARGV
  end
end

def attempt_create_security_group(ec2_client, group_name, description, vpc_id)
  puts 'Attempting to create security group...'
  security_group_id = create_security_group(ec2_client, group_name, description,
  vpc_id)
  puts 'Could not create security group. Skipping this step.' if
  security_group_id == 'Error'
  security_group_id
end

def add_inbound_rules(ec2_client, security_group_id, ip_protocol, from_port,
  to_port, cidr_ip_range)
  puts 'Attempting to add inbound rules to security group...'
  return if security_group_ingress_authorized?(ec2_client, security_group_id,
  ip_protocol, from_port, to_port,
  cidr_ip_range)

  puts 'Could not add inbound rule to security group. Skipping this step.'
end

def attempt_delete_security_group(ec2_client, security_group_id)
  puts "\nAttempting to delete security group..."
  return if security_group_deleted?(ec2_client, security_group_id)

  puts 'Could not delete security group. You must delete it yourself.'
end

def display_help
  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"
end

def default_values
  [
    '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'
  ]
end

run_me if $PROGRAM_NAME == __FILE__

```

- Para API obter detalhes, consulte [CreateSecurityGroup](#) em AWS SDK for Ruby API Referência.

Rust

SDK para Rust

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```

pub async fn create_security_group(
    &self,
    name: &str,
    description: &str,
) -> Result<SecurityGroup, EC2Error> {
    tracing::info!("Creating security group {name}");
    let create_output = self
        .client
        .create_security_group()
        .group_name(name)

```

```

        .description(description)
        .send()
        .await
        .map_err(EC2Error::from)?;

    let group_id = create_output
        .group_id
        .ok_or_else(|| EC2Error::new("Missing security group id after
creation"))?;

    let group = self
        .describe_security_group(&group_id)
        .await?
        .ok_or_else(|| {
            EC2Error::new(format!("Could not find security group with id
{group_id}"))
        })?;

    tracing::info!("Created security group {name} as {group_id}");

    Ok(group)
}

```

- Para API obter detalhes, consulte [CreateSecurityGroup](#) AWS SDKAPIreferência do Rust.

SAP ABAP

SDKpara SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```

TRY.
    oo_result = lo_ec2->createsecuritygroup(
        iv_description = 'Security group example'
        iv_groupname = iv_security_group_name
        iv_vpcid = iv_vpc_id
    )
    " oo_result is
returned for testing purposes. "

```

```
    ).
    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'.
  ENDTTRY.
```

- Para API obter detalhes, consulte [CreateSecurityGroup AWS SDK](#) para SAP ABAP API referência.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **CreateSnapshot** com um CLI

Os exemplos de código a seguir mostram como usar o CreateSnapshot.

CLI

AWS CLI

Para criar um instantâneo

Esse exemplo de comando cria um instantâneo do volume com um ID do volume `vol-1234567890abcdef0` e uma breve descrição para identificar o instantâneo.

Comando:

```
aws ec2 create-snapshot --volume-id vol-1234567890abcdef0 --description "This is my root volume snapshot"
```

Saída:

```
{
  "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"
}
```

Para criar um instantâneo com tags

Esse exemplo de comando cria um instantâneo e aplica duas tags: `purpose=prod` e `costcenter=123`.

Comando:

```
aws ec2 create-snapshot --volume-id vol-1234567890abcdef0
--description 'Prod backup' --tag-specifications
'ResourceType=snapshot,Tags=[{Key=purpose,Value=prod},
{Key=costcenter,Value=123}]'
```

Saída:

```
{
  "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"
}
```

- Para API obter detalhes, consulte [CreateSnapshot](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo cria um instantâneo do volume especificado.

```
New-EC2Snapshot -VolumeId vol-12345678 -Description "This is a test"
```

Saída:

```
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
```

- Para API obter detalhes, consulte [CreateSnapshot](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **CreateSpotDatafeedSubscription** com um CLI

Os exemplos de código a seguir mostram como usar o `CreateSpotDatafeedSubscription`.

CLI

AWS CLI

Para criar um feed de dados da Instância Spot

O `create-spot-datafeed-subscription` exemplo a seguir cria um feed de dados da Instância Spot.

```
aws ec2 create-spot-datafeed-subscription \  
  --bucket my-bucket \  
  --prefix spot-data-feed
```

Saída:

```
{  
  "SpotDatafeedSubscription": {  
    "Bucket": "my-bucket",  
    "OwnerId": "123456789012",  
    "Prefix": "spot-data-feed",  
    "State": "Active"  
  }  
}
```

O feed de dados é armazenado no bucket do Amazon S3 que você especificou. Os nomes dos arquivos desse feed de dados têm o seguinte formato.

```
my-bucket.s3.amazonaws.com/spot-data-feed/123456789012.YYYY-MM-DD-  
HH.n.abcd1234.gz
```

Para obter mais informações, consulte o [feed de dados da instância spot](#) no Guia do usuário do Amazon Elastic Compute Cloud para instâncias Linux.

- Para API obter detalhes, consulte [CreateSpotDatafeedSubscription](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Esse exemplo cria um feed de dados da instância spot.

```
New-EC2SpotDatafeedSubscription -Bucket amzn-s3-demo-bucket -Prefix spotdata
```

Saída:

```
Bucket   : my-s3-bucket
Fault    :
OwnerId  : 123456789012
Prefix   : spotdata
State    : Active
```

- Para API obter detalhes, consulte [CreateSpotDatafeedSubscription](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **CreateSubnet** com um AWS SDK ou CLI

Os exemplos de código a seguir mostram como usar o CreateSubnet.

CLI

AWS CLI

Exemplo 1: Para criar uma sub-rede somente com um IPv4 CIDR bloco

O create-subnet exemplo a seguir cria uma sub-rede no bloco especificado VPC com o IPv4 CIDR bloco especificado.

```
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}]
```

Saída:

```
{
  "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"
  }
}

```

Exemplo 2: Para criar uma sub-rede com blocos IPv4 e IPv6 CIDR

O `create-subnet` exemplo a seguir cria uma sub-rede no especificado VPC com os IPv6 CIDR blocos IPv4 especificados.

```

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}]

```

Saída:

```

{
  "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"
  }
}

```

Exemplo 3: Para criar uma sub-rede somente com um IPv6 CIDR bloco

O `create-subnet` exemplo a seguir cria uma sub-rede no bloco especificado VPC com o IPv6 CIDR bloco especificado.

```

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}]

```

Saída:

```

{
  "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"
  }
}

```

Para obter mais informações, consulte [VPCse sub-redes no Guia VPC](#) do usuário da Amazon.

- Para API obter detalhes, consulte [CreateSubnet](#)na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo cria uma sub-rede com o especificadoCIDR.

```
New-EC2Subnet -VpcId vpc-12345678 -CidrBlock 10.0.0.0/24
```

Saída:

```
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
```

- Para API obter detalhes, consulte [CreateSubnet](#) em Referência de AWS Tools for PowerShell cmdlet.

Ruby

SDK para Ruby

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
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}'."
  true
rescue StandardError => e
  puts "Error creating or tagging subnet: #{e.message}"
  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
end

run_me if $PROGRAM_NAME == __FILE__

```

- Para API obter detalhes, consulte [CreateSubnet](#) em AWS SDK for Ruby API Referência.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **CreateTags** com um AWS SDK ou CLI

Os exemplos de código a seguir mostram como usar o CreateTags.

C++

SDK para C++

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```

//! Add or overwrite only the specified tags for the specified Amazon Elastic
  Compute Cloud (Amazon EC2) resource or resources.
/*!
  \param resources: The resources for the tags.
  \param tags: Vector of tags.
  \param clientConfiguration: AWS client configuration.
  \return bool: Function succeeded.
*/
bool AwsDoc::EC2::createTags(const Aws::Vector<Aws::String> &resources,
                             const Aws::Vector<Aws::EC2::Model::Tag> &tags,
                             const Aws::Client::ClientConfiguration
  &clientConfiguration) {

```

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

Aws::EC2::Model::CreateTagsRequest createTagsRequest;
createTagsRequest.SetResources(resources);
createTagsRequest.SetTags(tags);

Aws::EC2::Model::CreateTagsOutcome outcome =
ec2Client.CreateTags(createTagsRequest);

if (outcome.IsSuccess()) {
    std::cout << "Successfully created tags for resources" << std::endl;
} else {
    std::cerr << "Failed to create tags for resources, " <<
outcome.GetError().GetMessage() << std::endl;
}

return outcome.IsSuccess();
}
```

- Para API obter detalhes, consulte [CreateTags](#) em AWS SDK for C++ API Referência.

CLI

AWS CLI

Exemplo 1: Para adicionar uma tag a um recurso

O `create-tags` exemplo a seguir adiciona a tag `Stack=production` à imagem especificada ou substitui uma tag existente por AMI onde está `Stack` a chave da tag.

```
aws ec2 create-tags \
  --resources ami-1234567890abcdef0 \
  --tags Key=Stack,Value=production
```

Para obter mais informações, consulte [Este é o título do tópico](#) no Guia do usuário do Amazon Elastic Compute Cloud para instâncias Linux.

Exemplo 2: Para adicionar tags a vários recursos

O `create-tags` exemplo a seguir adiciona (ou substitui) duas tags para uma instância AMI e uma. Uma das tags tem uma chave (`webserver`), mas nenhum valor (o valor é definido como uma string vazia). A outra tag tem uma chave (`stack`) e um valor (`Production`).

```
aws ec2 create-tags \  
  --resources ami-1a2b3c4d i-1234567890abcdef0 \  
  --tags Key=webserver,Value= Key=stack,Value=Production
```

Para obter mais informações, consulte [Este é o título do tópico](#) no Guia do usuário do Amazon Elastic Compute Cloud para instâncias Linux.

Exemplo 3: Para adicionar tags contendo caracteres especiais

O exemplo `create-tags` a seguir adiciona a tag `[Group]=test` para uma instância. Os colchetes (`[` e `]`) são caracteres especiais e devem ser recuados. Os exemplos a seguir também usam o caractere de continuação de linha apropriado para cada ambiente.

Se você estiver usando o Windows, coloque o elemento com caracteres especiais entre aspas duplas (`"`) e preceda cada caractere de aspas duplas com uma barra invertida (`\`) da seguinte maneira:

```
aws ec2 create-tags ^  
  --resources i-1234567890abcdef0 ^  
  --tags Key="\[Group]",Value=test
```

Se você estiver usando o Windows PowerShell, coloque o elemento no valor que tem caracteres especiais com aspas duplas (`"`), preceda cada caractere de aspas duplas com uma barra invertida (`\`) e, em seguida, coloque toda a estrutura de chave e valor entre aspas simples (`'`) da seguinte forma:

```
aws ec2 create-tags `\  
  --resources i-1234567890abcdef0 `\  
  --tags 'Key="\[Group]",Value=test'
```

Se você estiver usando Linux ou OS X, coloque o elemento com caracteres especiais entre aspas duplas (`"`) e coloque toda a estrutura de chave e valor entre aspas simples (`'`) da seguinte maneira:

```
aws ec2 create-tags \  
  --resources i-1234567890abcdef0 \  
  --tags Key="[Group]",Value=test
```

```
--resources i-1234567890abcdef0 \  
--tags 'Key="[Group]",Value=test'
```

Para obter mais informações, consulte [Este é o título do tópico](#) no Guia do usuário do Amazon Elastic Compute Cloud para instâncias Linux.

- Para API obter detalhes, consulte [CreateTags](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo adiciona uma única tag ao recurso especificado. A chave da tag é 'myTag' e o valor da tag é 'myTagValue'. A sintaxe usada neste exemplo requer a PowerShell versão 3 ou superior.

```
New-EC2Tag -Resource i-12345678 -Tag @{ Key="myTag"; Value="myTagValue" }
```

Exemplo 2: Este exemplo atualiza ou adiciona as tags especificadas ao recurso especificado. A sintaxe usada neste exemplo requer a PowerShell versão 3 ou superior.

```
New-EC2Tag -Resource i-12345678 -Tag @( @{ Key="myTag"; Value="newTagValue" },  
@{ Key="test"; Value="anotherTagValue" } )
```

Exemplo 3: Com a PowerShell versão 2, você deve usar New-Object para criar a tag para o parâmetro Tag.

```
$tag = New-Object Amazon.EC2.Model.Tag  
$tag.Key = "myTag"  
$tag.Value = "myTagValue"  
  
New-EC2Tag -Resource i-12345678 -Tag $tag
```

- Para API obter detalhes, consulte [CreateTag](#) em Referência de AWS Tools for PowerShell cmdlet.

Rust

SDK para Rust

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

Este exemplo aplica a tag Name Depois de criar uma instância.

```
pub async fn create_instance<'a>(
    &self,
    image_id: &'a str,
    instance_type: InstanceType,
    key_pair: &'a KeyPairInfo,
    security_groups: Vec<&'a SecurityGroup>,
) -> Result<String, EC2Error> {
    let run_instances = self
        .client
        .run_instances()
        .image_id(image_id)
        .instance_type(instance_type)
        .key_name(
            key_pair
                .key_name()
                .ok_or_else(|| EC2Error::new("Missing key name when launching
instance"))?,
        )
        .set_security_group_ids(Some(
            security_groups
                .iter()
                .filter_map(|sg| sg.group_id.clone())
                .collect(),
        ))
        .min_count(1)
        .max_count(1)
        .send()
        .await?;

    if run_instances.instances().is_empty() {
        return Err(EC2Error::new("Failed to create instance"));
    }
}
```

```
    }

    let instance_id = run_instances.instances()[0].instance_id().unwrap();
    let response = self
        .client
        .create_tags()
        .resources(instance_id)
        .tags(
            Tag::builder()
                .key("Name")
                .value("From SDK Examples")
                .build(),
        )
        .send()
        .await;

    match response {
        Ok(_) => tracing::info!("Created {instance_id} and applied tags."),
        Err(err) => {
            tracing::info!("Error applying tags to {instance_id}: {err:?}");
            return Err(err.into());
        }
    }

    tracing::info!("Instance is created.");

    Ok(instance_id.to_string())
}
```

- Para API obter detalhes, consulte [CreateTags](#) a AWS SDK API referência do Rust.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **CreateVolume** com um CLI

Os exemplos de código a seguir mostram como usar o `CreateVolume`.

CLI

AWS CLI

Para criar um volume vazio de uso geral SSD (gp2)

O `create-volume` exemplo a seguir cria um volume de uso geral SSD (gp2) de 80 GiB na zona de disponibilidade especificada. Observe que a região atual deve ser `us-east-1`, ou você pode adicionar o `--region` parâmetro para especificar a região para o comando.

```
aws ec2 create-volume \  
  --volume-type gp2 \  
  --size 80 \  
  --availability-zone us-east-1a
```

Saída:

```
{  
  "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  
}
```

Se você não especificar um tipo de volume, o tipo de volume padrão será `gp2`.

```
aws ec2 create-volume \  
  --size 80 \  
  --availability-zone us-east-1a
```

Exemplo 2: Para criar um volume provisionado IOPS SSD (io1) a partir de um snapshot

O `create-volume` exemplo a seguir cria um volume provisionado IOPS SSD (io1) com 1000 provisionados IOPS na zona de disponibilidade especificada usando o snapshot especificado.

```
aws ec2 create-volume \  
  --size 80 \  
  --availability-zone us-east-1a
```



```
--volume-type io1 \  
--iops 1000 \  
--snapshot-id snap-066877671789bd71b \  
--availability-zone us-east-1a
```

Saída:

```
{  
  "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  
}
```

Exemplo 3: Para criar um volume criptografado

O `create-volume` exemplo a seguir cria um volume criptografado usando o padrão CMK para EBS criptografia. Se a criptografia por padrão estiver desativada, você deverá especificar o `--encrypted` parâmetro da seguinte forma.

```
aws ec2 create-volume \  
  --size 80 \  
  --encrypted \  
  --availability-zone us-east-1a
```

Saída:

```
{  
  "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  
}
```

Se a criptografia por padrão estiver ativada, o comando de exemplo a seguir cria um volume criptografado, mesmo sem o `--encrypted` parâmetro.

```
aws ec2 create-volume \  
  --size 80 \  
  --availability-zone us-east-1a
```

Se você usar o `--kms-key-id` parâmetro para especificar um cliente gerenciado CMK, deverá especificar o `--encrypted` parâmetro mesmo se a criptografia, por padrão, estiver ativada.

```
aws ec2 create-volume \  
  --volume-type gp2 \  
  --size 80 \  
  --encrypted \  
  --kms-key-id 0ea3fef3-80a7-4778-9d8c-1c0c6EXAMPLE \  
  --availability-zone us-east-1a
```

Exemplo 4: Para criar um volume com tags

O `create-volume` exemplo a seguir cria um volume e adiciona duas tags.

```
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}]'
```

- Para API obter detalhes, consulte [CreateVolume](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo cria o volume especificado.

```
New-EC2Volume -Size 50 -AvailabilityZone us-west-2a -VolumeType gp2
```

Saída:

```
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
```

Exemplo 2: Esse exemplo de solicitação cria um volume e aplica uma tag com uma chave de pilha e um valor de produção.

```
$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
```

- Para API obter detalhes, consulte [CreateVolume](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **CreateVpc** com um AWS SDK ou CLI

Os exemplos de código a seguir mostram como usar o CreateVpc.

CLI

AWS CLI

Exemplo 1: Para criar um VPC

O `create-vpc` exemplo a seguir cria um VPC com o IPv4 CIDR bloco especificado e uma tag Name.

```
aws ec2 create-vpc \  
  --cidr-block 10.0.0.0/16 \  
  --tag-specifications ResourceType=vpc,Tags=[{Key=Name,Value=MyVpc}]
```

Saída:

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

Exemplo 2: Para criar um VPC com locação dedicada

O `create-vpc` exemplo a seguir cria um VPC com o IPv4 CIDR bloco especificado e a locação dedicada.

```
aws ec2 create-vpc \  
  --cidr-block 10.0.0.0/16 \  
  --instance-tenancy dedicated
```

Saída:

```
{  
  "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  
  }  
}
```

Exemplo 3: Para criar um VPC com um IPv6 CIDR bloco

O `create-vpc` exemplo a seguir cria um VPC com um bloco fornecido pela Amazon IPv6CIDR.

```
aws ec2 create-vpc \  
  --cidr-block 10.0.0.0/16 \  
  --amazon-provided-ipv6-cidr-block
```

Saída:

```
{
  "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
  }
}
```

Exemplo 4: Para criar um VPC com a CIDR a partir de um IPAM pool

O `create-vpc` exemplo a seguir cria um VPC com CIDR a de um pool do Amazon VPC IP Address Manager (IPAM).

Linux e 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"}]
```

Saída:

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

```
}  
}
```

Para obter mais informações, consulte [Criar um VPC que use um IPAM pool CIDR](#) no Guia do VPC IPAM usuário da Amazon.

- Para API obter detalhes, consulte [CreateVpc](#) na Referência de AWS CLI Comandos.

PHP

SDK para PHP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
/**  
 * @param string $cidr  
 * @return array  
 */  
public function createVpc(string $cidr): array  
{  
    try {  
        $result = $this->ec2Client->createVpc([  
            "CidrBlock" => $cidr,  
        ]);  
        return $result['Vpc'];  
    } catch (Ec2Exception $caught){  
        echo "There was a problem creating the VPC: {$caught->getAwsErrorMessage()}\n";  
        throw $caught;  
    }  
}
```

- Para API obter detalhes, consulte [CreateVpc](#) em AWS SDK for PHP API Referência.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo cria um VPC com o especificadoCIDR. A Amazon VPC também cria o seguinte paraVPC: um conjunto de DHCP opções padrão, uma tabela de rotas principal e uma rede padrãoACL.

```
New-EC2VPC -CidrBlock 10.0.0.0/16
```

Saída:

```
CidrBlock      : 10.0.0.0/16
DhcpOptionsId  : dopt-1a2b3c4d
InstanceTenancy : default
IsDefault      : False
State          : pending
Tags           : {}
VpcId          : vpc-12345678
```

- Para API obter detalhes, consulte [CreateVpcem](#) Referência de AWS Tools for PowerShell cmdlet.

Ruby

SDKpara Ruby

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
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}'."
  true
rescue StandardError => e
  puts e.message
  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__
```

- Para API obter detalhes, consulte [CreateVpc](#) em AWS SDK for Ruby API Referência.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **CreateVpcEndpoint** com um AWS SDK ou CLI

Os exemplos de código a seguir mostram como usar o CreateVpcEndpoint.

CLI

AWS CLI

Exemplo 1: Para criar um endpoint de gateway

O `create-vpc-endpoint` exemplo a seguir cria um VPC endpoint de gateway entre o Amazon S3 VPC `vpc-1a2b3c4d` e o Amazon S3 na região e `us-east-1` associa a `rtb-11aa22bb` tabela de rotas ao endpoint.

```
aws ec2 create-vpc-endpoint \  
  --vpc-id vpc-1a2b3c4d \  
  --service-name com.amazonaws.us-east-1.s3 \  
  --route-table-ids rtb-11aa22bb
```

Saída:

```
{  
  "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"  
  }  
}
```

Para obter mais informações, consulte [Criação de um endpoint de gateway](#) no AWS PrivateLink Guia.

Exemplo 2: Para criar um endpoint de interface

O `create-vpc-endpoint` exemplo a seguir cria um VPC endpoint de interface entre o Amazon S3 VPC `vpc-1a2b3c4d` e o Amazon S3 na `us-east-1` região. O comando cria o endpoint na sub-rede `subnet-1a2b3c4d`, o associa ao grupo `sg-1a2b3c4d` de segurança e adiciona uma tag com uma chave de "Serviço" e um valor de "S3".

```
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}]
```

Saída:

```
{
  "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"
}
}

```

Para obter mais informações, consulte [Criação de um endpoint de interface](#) no Guia do usuário do AWS PrivateLink

Exemplo 3: Para criar um endpoint do Gateway Load Balancer

O `create-vpc-endpoint` exemplo a seguir cria um endpoint do Gateway Load Balancer entre VPC `vpc-111122223333aabbcc` e um serviço configurado usando um Gateway Load Balancer.

```

aws ec2 create-vpc-endpoint \
  --service-name com.amazonaws.vpce.us-east-1.vpce-svc-123123a1c43abc123 \
  --vpc-endpoint-type GatewayLoadBalancer \
  --vpc-id vpc-111122223333aabbcc \
  --subnet-ids subnet-0011aabbcc2233445

```

Saída:

```

{
  "VpcEndpoint": {
    "VpcEndpointId": "vpce-aabbaabbaabbaabba",
    "VpcEndpointType": "GatewayLoadBalancer",

```

```
"VpcId": "vpc-111122223333aabb",
"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"
}
}
```

Para obter mais informações, consulte os [endpoints do Gateway Load Balancer no Guia](#) do usuário do. AWS PrivateLink

- Para API obter detalhes, consulte [CreateVpcEndpoint](#) na Referência de AWS CLI Comandos.

PHP

SDK para PHP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
/**
 * @param string $serviceName
 * @param string $vpcId
 * @param array $routeTableIds
 * @return array
 */
public function createVpcEndpoint(string $serviceName, string $vpcId, array
$routeTableIds): array
{
    try {
```

```

        $result = $this->ec2Client->createVpcEndpoint([
            'ServiceName' => $serviceName,
            'VpcId' => $vpcId,
            'RouteTableIds' => $routeTableIds,
        ]);

        return $result["VpcEndpoint"];
    } catch (Ec2Exception $caught){
        echo "There was a problem creating the VPC Endpoint: {"$caught->getAwsErrorMessage()}\n";
        throw $caught;
    }
}

```

- Para API obter detalhes, consulte [CreateVpcEndpoint](#) em AWS SDK for PHP API Referência.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo cria um novo VPC endpoint para o serviço `com.amazonaws.eu-west-1.s3` no `vpc-0fc1ff23f45b678eb` VPC

```

New-EC2VpcEndpoint -ServiceName com.amazonaws.eu-west-1.s3 -VpcId
vpc-0fc1ff23f45b678eb

```

Saída:

```

ClientToken VpcEndpoint
-----
                Amazon.EC2.Model.VpcEndpoint

```

- Para API obter detalhes, consulte [CreateVpcEndpoint](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use `CreateVpnConnection` com um CLI

Os exemplos de código a seguir mostram como usar o `CreateVpnConnection`.

CLI

AWS CLI

Exemplo 1: Para criar uma VPN conexão com roteamento dinâmico

O `create-vpn-connection` exemplo a seguir cria uma VPN conexão entre o gateway privado virtual especificado e o gateway do cliente especificado e aplica tags à VPN conexão. A saída inclui as informações de configuração do dispositivo de gateway do cliente, em XML formato.

```
aws ec2 create-vpn-connection \  
  --type ipsec.1 \  
  --customer-gateway-id cgw-001122334455aabb \  
  --vpn-gateway-id vgw-1a1a1a1a1a1a2b2b \  
  --tag-specification 'ResourceType=vpn-connection, Tags=[{Key=Name, Value=BGP-VPN}]'
```

Saída:

```
{  
  "VpnConnection": {  
    "CustomerGatewayConfiguration": "...configuration information...",  
    "CustomerGatewayId": "cgw-001122334455aabb",  
    "Category": "VPN",  
    "State": "pending",  
    "VpnConnectionId": "vpn-123123123123abcab",  
    "VpnGatewayId": "vgw-1a1a1a1a1a1a2b2b",  
    "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"
      }
    ]
  }
}

```

Para obter mais informações, consulte [Como AWS Site-to-Site VPN funciona](#) no Guia AWS Site-to-Site VPN do usuário.

Exemplo 2: Para criar uma VPN conexão com roteamento estático

O `create-vpn-connection` exemplo a seguir cria uma VPN conexão entre o gateway privado virtual especificado e o gateway do cliente especificado. As opções especificam o roteamento estático. A saída inclui as informações de configuração do dispositivo de gateway do cliente, em XML formato.

```

aws ec2 create-vpn-connection \
  --type ipsec.1 \
  --customer-gateway-id cgw-001122334455aabbcc \
  --vpn-gateway-id vgw-1a1a1a1a1a1a2b2b2 \
  --options "{\"StaticRoutesOnly\":true}"

```

Saída:

```

{
  "VpnConnection": {
    "CustomerGatewayConfiguration": "..configuration information...",
    "CustomerGatewayId": "cgw-001122334455aabbcc",
    "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": []
}
}

```

Para obter mais informações, consulte [Como AWS Site-to-Site VPN funciona](#) no Guia AWS Site-to-Site VPN do usuário.

Exemplo 3: Para criar uma VPN conexão e especificar sua própria chave interna CIDR e pré-compartilhada

O `create-vpn-connection` exemplo a seguir cria uma VPN conexão e especifica o CIDR bloco de endereço IP interno e uma chave pré-compartilhada personalizada para cada túnel. Os valores especificados são retornados nas `CustomerGatewayConfiguration` informações.

```

aws ec2 create-vpn-connection \
  --type ipsec.1 \
  --customer-gateway-id cgw-001122334455aabbcc \
  --vpn-gateway-id vgw-1a1a1a1a1a1a2b2b2 \
  --options
  TunnelOptions='[{TunnelInsideCidr=169.254.12.0/30,PreSharedKey=ExamplePreSharedKey1},
  {TunnelInsideCidr=169.254.13.0/30,PreSharedKey=ExamplePreSharedKey2}]'

```

Saída:

```

{
  "VpnConnection": {
    "CustomerGatewayConfiguration": "..configuration information...",
    "CustomerGatewayId": "cgw-001122334455aabbcc",
    "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": []
  }
}

```

Para obter mais informações, consulte [Como AWS Site-to-Site VPN funciona](#) no Guia AWS Site-to-Site VPN do usuário.

Exemplo 4: Para criar uma VPN conexão que ofereça suporte ao IPv6 tráfego

O `create-vpn-connection` exemplo a seguir cria uma VPN conexão que dá suporte ao IPv6 tráfego entre o gateway de trânsito especificado e o gateway do cliente especificado. As opções de túnel para ambos os túneis especificam que AWS deve iniciar a negociação. IKE

```

aws ec2 create-vpn-connection \
  --type ipsec.1 \
  --transit-gateway-id tgw-12312312312312312 \
  --customer-gateway-id cgw-001122334455aabbcc \
  --options TunnelInsideIpVersion=ipv6,TunnelOptions=[{StartupAction=start},
{StartupAction=start}]

```

Saída:

```

{
  "VpnConnection": {
    "CustomerGatewayConfiguration": "..configuration information...",
    "CustomerGatewayId": "cgw-001122334455aabbcc",

```

```

    "Category": "VPN",
    "State": "pending",
    "VpnConnectionId": "vpn-1111111112222222",
    "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": []
  }
}

```

Para obter mais informações, consulte [Como AWS Site-to-Site VPN funciona](#) no Guia AWS Site-to-Site VPN do usuário.

- Para API obter detalhes, consulte [CreateVpnConnection](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo cria uma VPN conexão entre o gateway privado virtual especificado e o gateway do cliente especificado. A saída inclui as informações de configuração que seu administrador de rede precisa, em XML formato.

```

New-EC2VpnConnection -Type ipsec.1 -CustomerGatewayId cgw-1a2b3c4d -VpnGatewayId
vgw-1a2b3c4d

```

Saída:

```
CustomerGatewayConfiguration : [XML document]
CustomerGatewayId           : cgw-1a2b3c4d
Options                     :
Routes                      : {}
State                       : pending
Tags                        : {}
Type                        :
VgwTelemetry                : {}
VpnConnectionId            : vpn-12345678
VpnGatewayId                : vgw-1a2b3c4d
```

Exemplo 2: Este exemplo cria a VPN conexão e captura a configuração em um arquivo com o nome especificado.

```
(New-EC2VpnConnection -CustomerGatewayId cgw-1a2b3c4d -VpnGatewayId
 vgw-1a2b3c4d).CustomerGatewayConfiguration | Out-File C:\path\vpn-
configuration.xml
```

Exemplo 3: Este exemplo cria uma VPN conexão, com roteamento estático, entre o gateway privado virtual especificado e o gateway do cliente especificado.

```
New-EC2VpnConnection -Type ipsec.1 -CustomerGatewayId cgw-1a2b3c4d -VpnGatewayId
 vgw-1a2b3c4d -Options_StaticRoutesOnly $true
```

- Para API obter detalhes, consulte [CreateVpnConnection](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **CreateVpnConnectionRoute** com um CLI

Os exemplos de código a seguir mostram como usar o `CreateVpnConnectionRoute`.

CLI

AWS CLI

Para criar uma rota estática para uma VPN conexão

Este exemplo cria uma rota estática para a VPN conexão especificada. Se o comando for bem-sucedido, nenhuma saída será retornada.

Comando:

```
aws ec2 create-vpn-connection-route --vpn-connection-id vpn-40f41529 --  
destination-cidr-block 11.12.0.0/16
```

- Para API obter detalhes, consulte [CreateVpnConnectionRoute](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo cria a rota estática especificada para a VPN conexão especificada.

```
New-EC2VpnConnectionRoute -VpnConnectionId vpn-12345678 -DestinationCidrBlock  
11.12.0.0/16
```

- Para API obter detalhes, consulte [CreateVpnConnectionRoute](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **CreateVpnGateway** com um CLI

Os exemplos de código a seguir mostram como usar o CreateVpnGateway.

CLI

AWS CLI

Para criar um gateway privado virtual

Este exemplo cria um gateway privado virtual.

Comando:

```
aws ec2 create-vpn-gateway --type ipsec.1
```

Saída:

```
{
  "VpnGateway": {
    "AmazonSideAsn": 64512,
    "State": "available",
    "Type": "ipsec.1",
    "VpnGatewayId": "vgw-9a4cacf3",
    "VpcAttachments": []
  }
}
```

Para criar um gateway privado virtual com um lado específico da Amazon ASN

Este exemplo cria um gateway privado virtual e especifica o Número do Sistema Autônomo (ASN) para o lado Amazon da BGP sessão.

Comando:

```
aws ec2 create-vpn-gateway --type ipsec.1 --amazon-side-asn 65001
```

Saída:

```
{
  "VpnGateway": {
    "AmazonSideAsn": 65001,
    "State": "available",
    "Type": "ipsec.1",
    "VpnGatewayId": "vgw-9a4cacf3",
  }
}
```



```
    "VpcAttachments": []  
  }  
}
```

- Para API obter detalhes, consulte [CreateVpnGateway](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo cria o gateway privado virtual especificado.

```
New-EC2VpnGateway -Type ipsec.1
```

Saída:

```
AvailabilityZone :  
State           : available  
Tags            : {}  
Type            : ipsec.1  
VpcAttachments  : {}  
VpnGatewayId   : vgw-1a2b3c4d
```

- Para API obter detalhes, consulte [CreateVpnGateway](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use `DeleteCustomerGateway` com um CLI

Os exemplos de código a seguir mostram como usar o `DeleteCustomerGateway`.

CLI

AWS CLI

Para excluir um gateway do cliente

Este exemplo exclui o gateway do cliente especificado. Se o comando for bem-sucedido, nenhuma saída será retornada.

Comando:

```
aws ec2 delete-customer-gateway --customer-gateway-id cgw-0e11f167
```

- Para API obter detalhes, consulte [DeleteCustomerGateway](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo exclui o gateway do cliente especificado. Você será solicitado a confirmar antes que a operação continue, a menos que você também especifique o parâmetro Force.

```
Remove-EC2CustomerGateway -CustomerGatewayId cgw-1a2b3c4d
```

Saída:

```
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"):
```

- Para API obter detalhes, consulte [DeleteCustomerGateway](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **DeleteDhcpOptions** com um CLI

Os exemplos de código a seguir mostram como usar o `DeleteDhcpOptions`.

CLI

AWS CLI

Para excluir um conjunto DHCP de opções

Este exemplo exclui o conjunto de DHCP opções especificado. Se o comando for bem-sucedido, nenhuma saída será retornada.

Comando:

```
aws ec2 delete-dhcp-options --dhcp-options-id dopt-d9070ebb
```

- Para API obter detalhes, consulte [DeleteDhcpOptions](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo exclui o conjunto de DHCP opções especificado. Você será solicitado a confirmar antes que a operação continue, a menos que você também especifique o parâmetro Force.

```
Remove-EC2DhcpOption -DhcpOptionsId dopt-1a2b3c4d
```

Saída:

```
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"):
```

- Para API obter detalhes, consulte [DeleteDhcpOption](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **DeleteFlowLogs** com um CLI

Os exemplos de código a seguir mostram como usar o DeleteFlowLogs.

CLI

AWS CLI

Para excluir um registro de fluxo

O delete-flow-logs exemplo a seguir exclui o registro de fluxo especificado.

```
aws ec2 delete-flow-logs --flow-log-id fl-11223344556677889
```

Saída:

```
{  
  "Unsuccessful": []  
}
```

- Para API obter detalhes, consulte [DeleteFlowLogs](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo remove o FlowLogId fl-01a2b3456a789c01 fornecido

```
Remove-EC2FlowLog -FlowLogId fl-01a2b3456a789c01
```

Saída:

```
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
```

- Para API obter detalhes, consulte [DeleteFlowLog](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use `DeleteInternetGateway` com um CLI

Os exemplos de código a seguir mostram como usar o `DeleteInternetGateway`.

CLI

AWS CLI

Para excluir um gateway de internet

O `delete-internet-gateway` exemplo a seguir exclui o gateway de internet especificado.

```
aws ec2 delete-internet-gateway \  
  --internet-gateway-id igw-0d0fb496b3EXAMPLE
```

Este comando não produz saída.

Para obter mais informações, consulte [Gateways de Internet](#) no Guia do VPC usuário da Amazon.

- Para API obter detalhes, consulte [DeleteInternetGateway](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo exclui o gateway de Internet especificado. Você será solicitado a confirmar antes que a operação continue, a menos que você também especifique o parâmetro `Force`.

```
Remove-EC2InternetGateway -InternetGatewayId igw-1a2b3c4d
```

Saída:

```
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"):
```

- Para API obter detalhes, consulte [DeleteInternetGateway](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **DeleteKeyPair** com um AWS SDK ou CLI

Os exemplos de código a seguir mostram como usar o `DeleteKeyPair`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação no contexto no seguinte exemplo de código:

- [Aprender os conceitos básicos](#)

.NET

AWS SDK for .NET

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
/// <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 (AmazonEC2Exception ec2Exception)
    {
        if (ec2Exception.ErrorCode == "InvalidKeyPair.NotFound")
        {
            _logger.LogError($"KeyPair {keyPairName} does not exist and
cannot be deleted. Please verify the key pair name and try again.");
        }

        return false;
    }
    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);
    }
}
```

- Para API obter detalhes, consulte [DeleteKeyPair](#) em AWS SDK for .NET API Referência.

Bash

AWS CLI com script Bash

 Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
#####
# 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 getopt 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 getopt "n:h" option; do
        case "${option}" in
            n) key_pair_name="${OPTARG}" ;;
            h)
                usage
                return 0
                ;;
        esac
    done
}
```



```

    \?)
    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
}

```

As funções utilitárias usadas neste exemplo.

```

#####
# 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
}

```

- Para API obter detalhes, consulte [DeleteKeyPair](#) na Referência de AWS CLI Comandos.

C++

SDK para C++

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```

//! Delete an Amazon Elastic Compute Cloud (Amazon EC2) instance key pair.

```

```

/!*
 \param keyPairName: A name for a key pair.
 \param clientConfiguration: AWS client configuration.
 \return bool: Function succeeded.
*/

bool AwsDoc::EC2::deleteKeyPair(const Aws::String &keyPairName,
                                const Aws::Client::ClientConfiguration
                                &clientConfiguration) {
    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;
    }

    return outcome.IsSuccess();
}

```

- Para API obter detalhes, consulte [DeleteKeyPair](#) em AWS SDK for C++ API Referência.

CLI

AWS CLI

Para excluir um par de chaves

O `delete-key-pair` exemplo a seguir exclui o par de chaves especificado.

```
aws ec2 delete-key-pair \
  --key-name my-key-pair
```

Saída:

```
{
  "Return": true,
  "KeyPairId": "key-03c8d3aceb53b507"
}
```

Para obter mais informações, consulte [Criar e excluir pares de chaves](#) no Guia do usuário da interface de linha de AWS comando.

- Para API obter detalhes, consulte [DeleteKeyPair](#) na Referência de AWS CLI Comandos.

Java

SDK para Java 2.x

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
/**
 * Deletes a key pair asynchronously.
 *
 * @param keyPair the name of the key pair to delete
 * @return a {@link CompletableFuture} that represents the result of the
 * asynchronous operation.
 *         The {@link CompletableFuture} will complete with a {@link
 * DeleteKeyPairResponse} object
 *         that provides the result of the key pair deletion operation.
 */
public CompletableFuture<DeleteKeyPairResponse> deleteKeysAsync(String
keyPair) {
    DeleteKeyPairRequest request = DeleteKeyPairRequest.builder()
        .keyName(keyPair)
        .build();

    // Initiate the asynchronous request to delete the key pair.
    CompletableFuture<DeleteKeyPairResponse> response =
getAsyncClient().deleteKeyPair(request);
    return response.whenComplete((resp, ex) -> {
        if (ex != null) {
```

```

        throw new RuntimeException("Failed to delete key pair: " +
keyPair, ex);
    } else if (resp == null) {
        throw new RuntimeException("No response received for deleting key
pair: " + keyPair);
    }
    });
}

```

- Para API obter detalhes, consulte [DeleteKeyPair](#) em AWS SDK for Java 2.x API Referência.

JavaScript

SDK para JavaScript (v3)

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```

import { DeleteKeyPairCommand, EC2Client } from "@aws-sdk/client-ec2";

/**
 * Deletes the specified key pair, by removing the public key from Amazon EC2.
 * @param {{ keyName: string }} options
 */
export const main = async ({ keyName }) => {
  const client = new EC2Client({});
  const command = new DeleteKeyPairCommand({
    KeyName: keyName,
  });

  try {
    await client.send(command);
    console.log("Successfully deleted key pair.");
  } catch (caught) {
    if (caught instanceof Error && caught.name === "MissingParameter") {
      console.warn(`${caught.message}. Did you provide the required value?`);
    } else {
      throw caught;
    }
  }
};

```

```
    }  
  }  
};
```

- Para API obter detalhes, consulte [DeleteKeyPair](#) em AWS SDK for JavaScript API Referência.

Kotlin

SDK para Kotlin

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
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")  
    }  
}
```

- Para API obter detalhes, consulte a [DeleteKeyPair](#) referência AWS SDK do Kotlin API.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo exclui o par de chaves especificado. Você será solicitado a confirmar antes que a operação continue, a menos que você também especifique o parâmetro Force.

```
Remove-EC2KeyPair -KeyName my-key-pair
```

Saída:

```
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"):
```

- Para API obter detalhes, consulte [DeleteKeyPair](#) em Referência de AWS Tools for PowerShell cmdlet.

Python

SDK para Python (Boto3)

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
class KeyPairWrapper:
    """
    Encapsulates Amazon Elastic Compute Cloud (Amazon EC2) key pair actions.
    This class provides methods to create, list, and delete EC2 key pairs.
    """

    def __init__(
        self,
        ec2_client: boto3.client,
        key_file_dir: Union[tempfile.TemporaryDirectory, str],
        key_pair: Optional[dict] = None,
    ):
        """
        Initializes the KeyPairWrapper with the specified EC2 client, key file
        directory,
        and an optional key pair.
```

```

        :param ec2_client: A Boto3 Amazon EC2 client. This client provides low-
level
                        access to AWS EC2 services.
        :param key_file_dir: The folder where the private key information is
stored.
                        This should be a secure folder.
        :param key_pair: A dictionary representing the Boto3 KeyPair object.
                        This is a high-level object that wraps key pair actions.
Optional.
    """
    self.ec2_client = ec2_client
    self.key_pair = key_pair
    self.key_file_path: Optional[str] = None
    self.key_file_dir = key_file_dir

    @classmethod
    def from_client(cls) -> "KeyPairWrapper":
        """
        Class method to create an instance of KeyPairWrapper using a new EC2
client
and a temporary directory for storing key files.

        :return: An instance of KeyPairWrapper.
        """
        ec2_client = boto3.client("ec2")
        return cls(ec2_client, tempfile.TemporaryDirectory())

    def delete(self, key_name: str) -> bool:
        """
        Deletes a key pair by its name.

        :param key_name: The name of the key pair to delete.
        :return: A boolean indicating whether the deletion was successful.
        :raises ClientError: If there is an error in deleting the key pair, for
example,
                        if the key pair does not exist.
        """
        try:
            self.ec2_client.delete_key_pair(KeyName=key_name)
            logger.info(f"Successfully deleted key pair: {key_name}")
            self.key_pair = None
            return True
        except self.ec2_client.exceptions.ClientError as err:

```



```

        logger.error(f"Deletion failed for key pair: {key_name}")
        error_code = err.response["Error"]["Code"]
        if error_code == "InvalidKeyPair.NotFound":
            logger.error(
                f"The key pair '{key_name}' does not exist and cannot be
deleted. "
                "Please verify the key pair name and try again."
            )
        raise

```

- Para API obter detalhes, consulte a [DeleteKeyPair](#) Referência AWS SDK do Python (Boto3). API

Rust

SDK para Rust

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

Envolva `delete_key` que também remove a chave privada de apoio. PEM

```

pub async fn delete(self, ec2: &EC2, util: &Util) -> Result<(), EC2Error> {
    if let Some(key_pair_id) = self.key_pair.key_pair_id() {
        ec2.delete_key_pair(key_pair_id).await?;
        if let Some(key_path) = self.key_file_path() {
            if let Err(err) = util.remove(key_path) {
                eprintln!("Failed to remove {key_path:?} ({err:?})");
            }
        }
    }
    Ok(())
}

```

```

pub async fn delete_key_pair(&self, key_pair_id: &str) -> Result<(),
EC2Error> {
    let key_pair_id: String = key_pair_id.into();
    tracing::info!("Deleting key pair {key_pair_id}");
    self.client
        .delete_key_pair()
        .key_pair_id(key_pair_id)
        .send()
        .await?;
    Ok(())
}

```

- Para API obter detalhes, consulte [DeleteKeyPaira AWS SDKAPIreferência do Rust](#).

SAP ABAP

SDKpara SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```

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.

```

- Para API obter detalhes, consulte [DeleteKeyPair AWSSDKpara SAP ABAP API referência](#).

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use `DeleteLaunchTemplate` com um AWS SDK ou CLI

Os exemplos de código a seguir mostram como usar o `DeleteLaunchTemplate`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação no contexto no seguinte exemplo de código:

- [Criar e gerenciar um serviço resiliente](#)

.NET

AWS SDK for .NET

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
/// <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 (AmazonEC2Exception ec2Exception)
    {
        if (ec2Exception.ErrorCode ==
            "InvalidLaunchTemplateName.NotFoundException")
        {
            _logger.LogError(
                $"Could not delete the template, the name
                {_launchTemplateName} was not found.");
        }
    }
}
```

```
        }  
  
        throw;  
    }  
    catch (Exception ex)  
    {  
        _logger.LogError($"An error occurred while deleting the template.:  
{ex.Message}");  
        throw;  
    }  
}
```

- Para API obter detalhes, consulte [DeleteLaunchTemplate](#) em AWS SDK for .NET API Referência.

CLI

AWS CLI

Para excluir um modelo de execução

Este exemplo exclui o modelo de execução especificado.

Comando:

```
aws ec2 delete-launch-template --launch-template-id lt-0abcd290751193123
```

Saída:

```
{  
  "LaunchTemplate": {  
    "LatestVersionNumber": 2,  
    "LaunchTemplateId": "lt-0abcd290751193123",  
    "LaunchTemplateName": "TestTemplate",  
    "DefaultVersionNumber": 2,  
    "CreatedBy": "arn:aws:iam::123456789012:root",  
    "CreateTime": "2017-11-23T16:46:25.000Z"  
  }  
}
```

- Para API obter detalhes, consulte [DeleteLaunchTemplate](#) na Referência de AWS CLI Comandos.

JavaScript

SDK para JavaScript (v3)

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
await client.send(  
  new DeleteLaunchTemplateCommand({  
    LaunchTemplateName: NAMES.launchTemplateName,  
  }),  
);
```

- Para API obter detalhes, consulte [DeleteLaunchTemplate](#) em AWS SDK for JavaScript API Referência.

Python

SDK para Python (Boto3)

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
class AutoScalingWrapper:  
    """  
    Encapsulates Amazon EC2 Auto Scaling and EC2 management actions.  
    """  
  
    def __init__(
```

```

self,
resource_prefix: str,
inst_type: str,
ami_param: str,
autoscaling_client: boto3.client,
ec2_client: boto3.client,
ssm_client: boto3.client,
iam_client: boto3.client,
):
    """
    Initializes the AutoScaler class with the necessary parameters.

    :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
    sts_client = boto3.client("sts")
    self.account_id = sts_client.get_caller_identity()["Account"]

    self.key_pair_name = f"{resource_prefix}-key-pair"
    self.launch_template_name = f"{resource_prefix}-template-"
    self.group_name = f"{resource_prefix}-group"

    # Happy path
    self.instance_policy_name = f"{resource_prefix}-pol"
    self.instance_role_name = f"{resource_prefix}-role"
    self.instance_profile_name = f"{resource_prefix}-prof"

    # Failure mode
    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"

```

```
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,
            )
        log.error(f"Full error:\n\t{err}")
```

- Para API obter detalhes, consulte a [DeleteLaunchTemplate](#) Referência AWS SDK do Python (Boto3). API

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **DeleteNetworkAcl** com um CLI

Os exemplos de código a seguir mostram como usar o `DeleteNetworkAcl`.

CLI

AWS CLI

Para excluir uma rede ACL

Este exemplo exclui a rede ACL especificada. Se o comando for bem-sucedido, nenhuma saída será retornada.

Comando:

```
aws ec2 delete-network-acl --network-acl-id acl-5fb85d36
```

- Para API obter detalhes, consulte [DeleteNetworkAcl](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo exclui a rede ACL especificada. Você será solicitado a confirmar antes que a operação continue, a menos que você também especifique o parâmetro Force.

```
Remove-EC2NetworkAcl -NetworkAclId acl-12345678
```

Saída:

```
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"):
```

- Para API obter detalhes, consulte [DeleteNetworkAcl](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use `DeleteNetworkAclEntry` com um CLI

Os exemplos de código a seguir mostram como usar o `DeleteNetworkAclEntry`.

CLI

AWS CLI

Para excluir uma ACL entrada de rede

Este exemplo exclui a regra de entrada número 100 da rede especificada. ACL Se o comando for bem-sucedido, nenhuma saída será retornada.

Comando:

```
aws ec2 delete-network-acl-entry --network-acl-id acl-5fb85d36 --ingress --rule-number 100
```

- Para API obter detalhes, consulte [DeleteNetworkAclEntry](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo remove a regra especificada da rede especificadaACL. Você será solicitado a confirmar antes que a operação continue, a menos que você também especifique o parâmetro `Force`.

```
Remove-EC2NetworkAclEntry -NetworkAclId acl-12345678 -Egress $false -RuleNumber 100
```

Saída:

```
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"):
```

- Para API obter detalhes, consulte [DeleteNetworkAclEntry](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **DeleteNetworkInterface** com um CLI

Os exemplos de código a seguir mostram como usar o `DeleteNetworkInterface`.

CLI

AWS CLI

Para excluir uma interface de rede

Este exemplo exclui a interface de rede especificada. Se o comando for bem-sucedido, nenhuma saída será retornada.

Comando:

```
aws ec2 delete-network-interface --network-interface-id eni-e5aa89a3
```

- Para API obter detalhes, consulte [DeleteNetworkInterface](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo exclui a interface de rede especificada. Você será solicitado a confirmar antes que a operação continue, a menos que você também especifique o parâmetro `Force`.

```
Remove-EC2NetworkInterface -NetworkInterfaceId eni-12345678
```

Saída:

```
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"):
```

- Para API obter detalhes, consulte [DeleteNetworkInterface](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **DeletePlacementGroup** com um CLI

Os exemplos de código a seguir mostram como usar o `DeletePlacementGroup`.

CLI

AWS CLI

Para excluir um grupo de posicionamento

Esse exemplo de comando exclui o grupo de posicionamento especificado.

Comando:

```
aws ec2 delete-placement-group --group-name my-cluster
```

- Para API obter detalhes, consulte [DeletePlacementGroup](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo exclui o grupo de posicionamento especificado. Você será solicitado a confirmar antes que a operação continue, a menos que você também especifique o parâmetro `Force`.

```
Remove-EC2PlacementGroup -GroupName my-placement-group
```

Saída:

```
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"):
```

- Para API obter detalhes, consulte [DeletePlacementGroup](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **DeleteRoute** com um CLI

Os exemplos de código a seguir mostram como usar o DeleteRoute.

CLI

AWS CLI

Para excluir uma rota

Este exemplo exclui a rota especificada da tabela de rotas especificada. Se o comando for bem-sucedido, nenhuma saída será retornada.

Comando:

```
aws ec2 delete-route --route-table-id rtb-22574640 --destination-cidr-  
block 0.0.0.0/0
```

- Para API obter detalhes, consulte [DeleteRoute](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo exclui a rota especificada da tabela de rotas especificada. Você será solicitado a confirmar antes que a operação continue, a menos que você também especifique o parâmetro Force.

```
Remove-EC2Route -RouteTableId rtb-1a2b3c4d -DestinationCidrBlock 0.0.0.0/0
```

Saída:

```
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"):
```

- Para API obter detalhes, consulte [DeleteRoute](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **DeleteRouteTable** com um CLI

Os exemplos de código a seguir mostram como usar o DeleteRouteTable.

CLI

AWS CLI

Para excluir uma tabela de rotas

Este exemplo exclui a tabela de rotas especificada. Se o comando for bem-sucedido, nenhuma saída será retornada.

Comando:

```
aws ec2 delete-route-table --route-table-id rtb-22574640
```

- Para API obter detalhes, consulte [DeleteRouteTable](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo exclui a tabela de rotas especificada. Você será solicitado a confirmar antes que a operação continue, a menos que você também especifique o parâmetro Force.

```
Remove-EC2RouteTable -RouteTableId rtb-1a2b3c4d
```

Saída:

```
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"):
```

- Para API obter detalhes, consulte [DeleteRouteTable](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **DeleteSecurityGroup** com um AWS SDK ou CLI

Os exemplos de código a seguir mostram como usar o DeleteSecurityGroup.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação no contexto no seguinte exemplo de código:

- [Aprender os conceitos básicos](#)

.NET

AWS SDK for .NET

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
/// <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)
{
    try
    {
        var response =
            await _amazonEC2.DeleteSecurityGroupAsync(
                new DeleteSecurityGroupRequest { GroupId = groupId });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }
    catch (AmazonEC2Exception ec2Exception)
    {
        if (ec2Exception.ErrorCode == "InvalidGroup.NotFound")
        {
            _logger.LogError(
                $"Security Group {groupId} does not exist and cannot be
deleted. Please verify the ID and try again.");
        }

        return false;
    }
    catch (Exception ex)
    {
        Console.WriteLine($"Couldn't delete the security group because:
{ex.Message}");
        return false;
    }
}
```

- Para API obter detalhes, consulte [DeleteSecurityGroup](#) em AWS SDK for .NET API Referência.

Bash

AWS CLI com script Bash

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
#####
# 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 getopt 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
}

```

As funções utilitárias usadas neste exemplo.

```

#####
# 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
}
```

- Para API obter detalhes, consulte [DeleteSecurityGroup](#) na Referência de AWS CLI Comandos.

C++

SDK para C++

 Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
#!/ Delete a security group.
/*!
  \param securityGroupID: A security group ID.
  \param clientConfiguration: AWS client configuration.
  \return bool: Function succeeded.
 */
bool AwsDoc::EC2::deleteSecurityGroup(const Aws::String &securityGroupID,
                                     const Aws::Client::ClientConfiguration
                                     &clientConfiguration) {
    Aws::EC2::EC2Client ec2Client(clientConfiguration);
    Aws::EC2::Model::DeleteSecurityGroupRequest request;

    request.SetGroupId(securityGroupID);
    Aws::EC2::Model::DeleteSecurityGroupOutcome 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;
    }

    return outcome.IsSuccess();
}
```

- Para API obter detalhes, consulte [DeleteSecurityGroup](#) em AWS SDK for C++ API Referência.

CLI

AWS CLI

[EC2-Clássico] Para excluir um grupo de segurança

Este exemplo exclui o grupo de segurança chamado `MySecurityGroup`. Se o comando for bem-sucedido, nenhuma saída será retornada.

Comando:

```
aws ec2 delete-security-group --group-name MySecurityGroup
```

[EC2-VPC] Para excluir um grupo de segurança

Este exemplo exclui o grupo de segurança com o ID `sg-903004f8`. Observe que você não pode referenciar um grupo de segurança para EC2 - VPC por nome. Se o comando for bem-sucedido, nenhuma saída será retornada.

Comando:

```
aws ec2 delete-security-group --group-id sg-903004f8
```

Para obter mais informações, consulte [Using Security Groups](#) no Guia do usuário da AWS Command Line Interface.

- Para API obter detalhes, consulte [DeleteSecurityGroup](#) na Referência de AWS CLI Comandos.

Java

SDK para Java 2.x

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
/**  
 * Deletes an EC2 security group asynchronously. */
```

```

    *
    * @param groupId the ID of the security group to delete
    * @return a CompletableFuture that completes when the security group is
    deleted
    */
    public CompletableFuture<Void> deleteEC2SecGroupAsync(String groupId) {
        DeleteSecurityGroupRequest request = DeleteSecurityGroupRequest.builder()
            .groupId(groupId)
            .build();

        CompletableFuture<DeleteSecurityGroupResponse> response =
getAsyncClient().deleteSecurityGroup(request);
        return response.whenComplete((resp, ex) -> {
            if (ex != null) {
                throw new RuntimeException("Failed to delete security group with
Id " + groupId, ex);
            } else if (resp == null) {
                throw new RuntimeException("No response received for deleting
security group with Id " + groupId);
            }
        }).thenApply(resp -> null);
    }

```

- Para API obter detalhes, consulte [DeleteSecurityGroup](#) em AWS SDK for Java 2.x API Referência.

JavaScript

SDK para JavaScript (v3)

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```

import { DeleteSecurityGroupCommand, EC2Client } from "@aws-sdk/client-ec2";

/**
 * Deletes a security group.

```

```
* @param {{ groupId: string }} options
*/
export const main = async ({ groupId }) => {
  const client = new EC2Client({});
  const command = new DeleteSecurityGroupCommand({
    GroupId: groupId,
  });

  try {
    await client.send(command);
    console.log("Security group deleted successfully.");
  } catch (caught) {
    if (caught instanceof Error && caught.name === "InvalidGroupId.Malformed") {
      console.warn(`${caught.message}. Please provide a valid GroupId.`);
    } else {
      throw caught;
    }
  }
};
```

- Para API obter detalhes, consulte [DeleteSecurityGroup](#) em AWS SDK for JavaScript API Referência.

Kotlin

SDK para Kotlin

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
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")
    }
}
```

- Para API obter detalhes, consulte a [DeleteSecurityGroup](#) referência AWS SDKdo Kotlin API.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo exclui o grupo de segurança especificado para EC2 -VPC. Você será solicitado a confirmar antes que a operação continue, a menos que você também especifique o parâmetro Force.

```
Remove-EC2SecurityGroup -GroupId sg-12345678
```

Saída:

```
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"):
```

Exemplo 2: Este exemplo exclui o grupo de segurança especificado para EC2 -Classic.

```
Remove-EC2SecurityGroup -GroupName my-security-group -Force
```

- Para API obter detalhes, consulte [DeleteSecurityGroup](#) em Referência de AWS Tools for PowerShell cmdlet.

Python

SDK para Python (Boto3)

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

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

    def __init__(self, ec2_client: boto3.client, security_group: Optional[str] =
    None):
        """
        Initializes the SecurityGroupWrapper with an EC2 client and an optional
        security group ID.

        :param ec2_client: A Boto3 Amazon EC2 client. This client provides low-
        level
                           access to AWS EC2 services.
        :param security_group: The ID of a security group to manage. This is a
        high-level identifier
                           that represents the security group.
        """
        self.ec2_client = ec2_client
        self.security_group = security_group

    @classmethod
    def from_client(cls) -> "SecurityGroupWrapper":
        """
        Creates a SecurityGroupWrapper instance with a default EC2 client.

        :return: An instance of SecurityGroupWrapper initialized with the default
        EC2 client.
        """
        ec2_client = boto3.client("ec2")
        return cls(ec2_client)
```



```
def delete(self, security_group_id: str) -> bool:
    """
    Deletes the specified security group.

    :param security_group_id: The ID of the security group to delete.
    Required.

    :returns: True if the deletion is successful.
    :raises ClientError: If the security group cannot be deleted due to an
    AWS service error.
    """
    try:
        self.ec2_client.delete_security_group(GroupId=security_group_id)
        logger.info(f"Successfully deleted security group
        '{security_group_id}'")
        return True
    except ClientError as err:
        logger.error(f"Deletion failed for security group
        '{security_group_id}'")
        error_code = err.response["Error"]["Code"]

        if error_code == "InvalidGroup.NotFound":
            logger.error(
                f"Security group '{security_group_id}' cannot be deleted
                because it does not exist."
            )
        elif error_code == "DependencyViolation":
            logger.error(
                f"Security group '{security_group_id}' cannot be deleted
                because it is still in use."
                " Verify that it is:"
                "\n\t- Detached from resources"
                "\n\t- Removed from references in other groups"
                "\n\t- Removed from VPC's as a default group"
            )
        raise
```

- Para API obter detalhes, consulte a [DeleteSecurityGroup](#) Referência AWS SDK do Python (Boto3). API

Rust

SDK para Rust

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
pub async fn delete_security_group(&self, group_id: &str) -> Result<(),
EC2Error> {
    tracing::info!("Deleting security group {group_id}");
    self.client
        .delete_security_group()
        .group_id(group_id)
        .send()
        .await?;
    Ok(())
}
```

- Para API obter detalhes, consulte [DeleteSecurityGroup](#) AWS SDK API referência do Rust.

SAP ABAP

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
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.
```

- Para API obter detalhes, consulte [DeleteSecurityGroup AWS SDK](#) para SAP ABAP API referência.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **DeleteSnapshot** com um AWS SDK ou CLI

Os exemplos de código a seguir mostram como usar o `DeleteSnapshot`.

CLI

AWS CLI

Para excluir um snapshot

Este exemplo de comando exclui um snapshot com o ID de snapshot de `snap-1234567890abcdef0`. Se o comando for bem-sucedido, nenhuma saída será retornada.

Comando:

```
aws ec2 delete-snapshot --snapshot-id snap-1234567890abcdef0
```

- Para API obter detalhes, consulte [DeleteSnapshot](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo exclui o instantâneo especificado. Você será solicitado a confirmar antes que a operação continue, a menos que você também especifique o parâmetro `Force`.

```
Remove-EC2Snapshot -SnapshotId snap-12345678
```

Saída:

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"):

- Para API obter detalhes, consulte [DeleteSnapshot](#) em Referência de AWS Tools for PowerShell cmdlet.

Rust**SDK para Rust****Note**

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
async fn delete_snapshot(client: &Client, id: &str) -> Result<(), Error> {
    client.delete_snapshot().snapshot_id(id).send().await?;

    println!("Deleted");

    Ok(())
}
```

- Para API obter detalhes, consulte [DeleteSnapshot](#) a AWS SDK API referência do Rust.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use `DeleteSpotDatafeedSubscription` com um CLI

Os exemplos de código a seguir mostram como usar o `DeleteSpotDatafeedSubscription`.

CLI

AWS CLI

Para cancelar uma assinatura de feed de dados da Spot Instance

Este exemplo de comando exclui uma assinatura de feed de dados Spot para a conta. Se o comando for bem-sucedido, nenhuma saída será retornada.

Comando:

```
aws ec2 delete-spot-datafeed-subscription
```

- Para API obter detalhes, consulte [DeleteSpotDatafeedSubscription](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Esse exemplo exclui o feed de dados da sua instância spot. Você será solicitado a confirmar antes que a operação continue, a menos que você também especifique o parâmetro Force.

```
Remove-EC2SpotDatafeedSubscription
```

Saída:

```
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"):
```

- Para API obter detalhes, consulte [DeleteSpotDatafeedSubscription](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use `DeleteSubnet` com um CLI

Os exemplos de código a seguir mostram como usar o `DeleteSubnet`.

CLI

AWS CLI

Para excluir uma sub-rede

Este exemplo exclui a sub-rede especificada. Se o comando for bem-sucedido, nenhuma saída será retornada.

Comando:

```
aws ec2 delete-subnet --subnet-id subnet-9d4a7b6c
```

- Para API obter detalhes, consulte [DeleteSubnet](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo exclui a sub-rede especificada. Você será solicitado a confirmar antes que a operação continue, a menos que você também especifique o parâmetro `Force`.

```
Remove-EC2Subnet -SubnetId subnet-1a2b3c4d
```

Saída:

```
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"):
```

- Para API obter detalhes, consulte [DeleteSubnet](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **DeleteTags** com um CLI

Os exemplos de código a seguir mostram como usar o DeleteTags.

CLI

AWS CLI

Exemplo 1: Para excluir uma tag de um recurso

O delete-tags exemplo a seguir exclui a tag Stack=Test da imagem especificada. Quando você especifica um valor e um nome de chave, a tag é excluída somente se o valor da tag corresponder ao valor especificado.

```
aws ec2 delete-tags \  
  --resources ami-1234567890abcdef0 \  
  --tags Key=Stack,Value=Test
```

É opcional especificar o valor de uma tag. O delete-tags exemplo a seguir exclui a tag com o nome purpose da chave da instância especificada, independentemente do valor da tag.

```
aws ec2 delete-tags \  
  --resources i-1234567890abcdef0 \  
  --tags Key=purpose
```

Se você especificar a string vazia como o valor da tag, a tag será excluída somente se o valor da tag for a string vazia. O delete-tags exemplo a seguir especifica a string vazia como o valor da tag a ser excluída.

```
aws ec2 delete-tags \  
  --resources i-1234567890abcdef0 \  
  --tags Key=Name,Value=
```

Exemplo 2: Para excluir uma tag de vários recursos

O `delete-tags` exemplo a seguir exclui a tag `purpose=test` de uma instância e de uma AMI Conforme mostrado no exemplo anterior, você pode omitir o valor da tag do comando.

```
aws ec2 delete-tags \  
  --resources i-1234567890abcdef0 ami-1234567890abcdef0 \  
  --tags Key=Purpose
```

- Para API obter detalhes, consulte [DeleteTags](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo exclui a tag especificada do recurso especificado, independentemente do valor da tag. A sintaxe usada neste exemplo requer a PowerShell versão 3 ou posterior.

```
Remove-EC2Tag -Resource i-12345678 -Tag @{ Key="myTag" } -Force
```

Exemplo 2: Este exemplo exclui a tag especificada do recurso especificado, mas somente se o valor da tag corresponder. A sintaxe usada neste exemplo requer a PowerShell versão 3 ou posterior.

```
Remove-EC2Tag -Resource i-12345678 -Tag @{ Key="myTag";Value="myTagValue" } -  
Force
```

Exemplo 3: Este exemplo exclui a tag especificada do recurso especificado, independentemente do valor da tag.

```
$tag = New-Object Amazon.EC2.Model.Tag  
$tag.Key = "myTag"  
  
Remove-EC2Tag -Resource i-12345678 -Tag $tag -Force
```

Exemplo 4: Este exemplo exclui a tag especificada do recurso especificado, mas somente se o valor da tag corresponder.


```
$tag = New-Object Amazon.EC2.Model.Tag
$tag.Key = "myTag"
$tag.Value = "myTagValue"

Remove-EC2Tag -Resource i-12345678 -Tag $tag -Force
```

- Para API obter detalhes, consulte [DeleteTags](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **DeleteVolume** com um CLI

Os exemplos de código a seguir mostram como usar o `DeleteVolume`.

CLI

AWS CLI

Para excluir um volume

Este exemplo de comando exclui um volume disponível com o ID do volume `devol-049df61146c4d7901`. Se o comando for bem-sucedido, nenhuma saída será retornada.

Comando:

```
aws ec2 delete-volume --volume-id vol-049df61146c4d7901
```

- Para API obter detalhes, consulte [DeleteVolume](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo separa o volume especificado. Você será solicitado a confirmar antes que a operação continue, a menos que você também especifique o parâmetro `Force`.

```
Remove-EC2Volume -VolumeId vol-12345678
```

Saída:

```
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"):
```

- Para API obter detalhes, consulte [DeleteVolume](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **DeleteVpc** com um AWS SDK ou CLI

Os exemplos de código a seguir mostram como usar o DeleteVpc.

CLI

AWS CLI

Para excluir um VPC

Este exemplo exclui o especificado VPC. Se o comando for bem-sucedido, nenhuma saída será retornada.

Comando:

```
aws ec2 delete-vpc --vpc-id vpc-a01106c2
```

- Para API obter detalhes, consulte [DeleteVpc](#) na Referência de AWS CLI Comandos.

PHP

SDK para PHP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
/**
 * @param string $vpcId
 * @return void
 */
public function deleteVpc(string $vpcId)
{
    try {
        $this->ec2Client->deleteVpc([
            "VpcId" => $vpcId,
        ]);
    } catch (Ec2Exception $caught){
        echo "There was a problem deleting the VPC: {$caught-
>getAwsErrorMessage()}\n";
        throw $caught;
    }
}
```

- Para API obter detalhes, consulte [DeleteVpc](#) em AWS SDK for PHP API Referência.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo exclui o especificado VPC. Você será solicitado a confirmar antes que a operação continue, a menos que você também especifique o parâmetro Force.

```
Remove-EC2Vpc -VpcId vpc-12345678
```

Saída:

```
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"):
```

- Para API obter detalhes, consulte [DeleteVpc](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use `DeleteVpcEndpoint` com um AWS SDK

O código de exemplo a seguir mostra como usar `DeleteVpcEndpoint`.

PHP

SDK para PHP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
/**
 * @param string $vpcEndpointId
 * @return void
 */
public function deleteVpcEndpoint(string $vpcEndpointId)
{
    try {
        $this->ec2Client->deleteVpcEndpoints([
            "VpcEndpointIds" => [$vpcEndpointId],
        ]);
    }
}
```

```
    }catch (Ec2Exception $caught){
        echo "There was a problem deleting the VPC Endpoint: {$caught-
>getAwsErrorMessage()}\n";
        throw $caught;
    }
}
```

- Para API obter detalhes, consulte [DeleteVpcEndpoint](#) em AWS SDK for PHP API Referência.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **DeleteVpnConnection** com um CLI

Os exemplos de código a seguir mostram como usar o `DeleteVpnConnection`.

CLI

AWS CLI

Para excluir uma VPN conexão

Este exemplo exclui a VPN conexão especificada. Se o comando for bem-sucedido, nenhuma saída será retornada.

Comando:

```
aws ec2 delete-vpn-connection --vpn-connection-id vpn-40f41529
```

- Para API obter detalhes, consulte [DeleteVpnConnection](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo exclui a VPN conexão especificada. Você será solicitado a confirmar antes que a operação continue, a menos que você também especifique o parâmetro `Force`.

```
Remove-EC2VpnConnection -VpnConnectionId vpn-12345678
```

Saída:

```
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"):
```

- Para API obter detalhes, consulte [DeleteVpnConnection](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use `DeleteVpnConnectionRoute` com um CLI

Os exemplos de código a seguir mostram como usar o `DeleteVpnConnectionRoute`.

CLI

AWS CLI

Para excluir uma rota estática de uma VPN conexão

Este exemplo exclui a rota estática especificada da VPN conexão especificada. Se o comando for bem-sucedido, nenhuma saída será retornada.

Comando:

```
aws ec2 delete-vpn-connection-route --vpn-connection-id vpn-40f41529 --
destination-cidr-block 11.12.0.0/16
```

- Para API obter detalhes, consulte [DeleteVpnConnectionRoute](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo remove a rota estática especificada da VPN conexão especificada. Você será solicitado a confirmar antes que a operação continue, a menos que você também especifique o parâmetro Force.

```
Remove-EC2VpnConnectionRoute -VpnConnectionId vpn-12345678 -DestinationCidrBlock 11.12.0.0/16
```

Saída:

```
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"):
```

- Para API obter detalhes, consulte [DeleteVpnConnectionRoute](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **DeleteVpnGateway** com um CLI

Os exemplos de código a seguir mostram como usar o `DeleteVpnGateway`.

CLI

AWS CLI

Para excluir um gateway privado virtual

Este exemplo exclui o gateway privado virtual especificado. Se o comando for bem-sucedido, nenhuma saída será retornada.

Comando:

```
aws ec2 delete-vpn-gateway --vpn-gateway-id vgw-9a4cacf3
```

- Para API obter detalhes, consulte [DeleteVpnGateway](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo exclui o gateway privado virtual especificado. Você será solicitado a confirmar antes que a operação continue, a menos que você também especifique o parâmetro Force.

```
Remove-EC2VpnGateway -VpnGatewayId vgw-1a2b3c4d
```

Saída:

```
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"):
```

- Para API obter detalhes, consulte [DeleteVpnGateway](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **DeregisterImage** com um CLI

Os exemplos de código a seguir mostram como usar o DeregisterImage.

CLI

AWS CLI

Para cancelar o registro de um AMI

Este exemplo cancela o registro especificado. AMI Se o comando for bem-sucedido, nenhuma saída será retornada.

Comando:

```
aws ec2 deregister-image --image-id ami-4fa54026
```

- Para API obter detalhes, consulte [DeregisterImage](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo cancela o registro especificado. AMI

```
Unregister-EC2Image -ImageId ami-12345678
```

- Para API obter detalhes, consulte [DeregisterImage](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **DescribeAccountAttributes** com um CLI

Os exemplos de código a seguir mostram como usar o DescribeAccountAttributes.

CLI

AWS CLI

Para descrever todos os atributos da sua AWS conta

Este exemplo descreve os atributos da sua AWS conta.

Comando:

```
aws ec2 describe-account-attributes
```

Saída:

```
{
  "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"
    }
  ]
}
]
}

```

Para descrever um único atributo para sua AWS conta

Este exemplo descreve o `supported-platforms` atributo da sua AWS conta.

Comando:

```
aws ec2 describe-account-attributes --attribute-names supported-platforms
```

Saída:

```

{
  "AccountAttributes": [
    {
      "AttributeName": "supported-platforms",
      "AttributeValues": [
        {
          "AttributeValue": "EC2"
        },
        {
          "AttributeValue": "VPC"
        }
      ]
    }
  ]
}

```

- Para API obter detalhes, consulte [DescribeAccountAttributes](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo descreve se você pode executar instâncias em EC2 -Classic e EC2 -VPC na região, ou somente em EC2 -VPC.

```
(Get-EC2AccountAttribute -AttributeName supported-platforms).AttributeValues
```

Saída:

```
AttributeValue
-----
EC2
VPC
```

Exemplo 2: Este exemplo descreve seu padrão VPC ou é “nenhum” se você não tiver um padrão VPC na região.

```
(Get-EC2AccountAttribute -AttributeName default-vpc).AttributeValues
```

Saída:

```
AttributeValue
-----
vpc-12345678
```

Exemplo 3: Este exemplo descreve o número máximo de instâncias sob demanda que você pode executar.

```
(Get-EC2AccountAttribute -AttributeName max-instances).AttributeValues
```

Saída:

```
AttributeValue
-----
20
```

- Para API obter detalhes, consulte [DescribeAccountAttributes](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **DescribeAddresses** com um AWS SDK ou CLI

Os exemplos de código a seguir mostram como usar o DescribeAddresses.

C++

SDK para C++

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
#!/ Describe all Elastic IP addresses.
/*!
 \param clientConfiguration: AWS client configuration.
 \return bool: Function succeeded.
 */
bool AwsDoc::EC2::describeAddresses(
    const Aws::Client::ClientConfiguration &clientConfiguration) {
    Aws::EC2::EC2Client ec2Client(clientConfiguration);
    Aws::EC2::Model::DescribeAddressesRequest request;
    Aws::EC2::Model::DescribeAddressesOutcome 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 Aws::Vector<Aws::EC2::Model::Address> &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;
}

return outcome.IsSuccess();
}
```

- Para API obter detalhes, consulte [DescribeAddresses](#) em AWS SDK for C++ API Referência.

CLI

AWS CLI

Exemplo 1: recuperar detalhes de todos os seus endereços IP elásticos

O exemplo `describe addresses` a seguir mostra os detalhes dos seus endereços IP elásticos.

```
aws ec2 describe-addresses
```

Saída:

```
{
  "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"
    }
]
}

```

Exemplo 2: Para recuperar detalhes de seus endereços IP elásticos para EC2 - VPC

O `describe-addresses` exemplo a seguir exibe detalhes sobre seus endereços IP elásticos para uso com instâncias em um VPC.

```

aws ec2 describe-addresses \
  --filters "Name=domain,Values=vpc"

```

Saída:

```

{
  "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"
    }
  ]
}

```

Exemplo 3: recuperar detalhes sobre um endereço IP elástico especificado pelo ID de alocação

O `describe-addresses` exemplo a seguir exibe detalhes sobre o endereço IP elástico com o ID de alocação especificado, que está associado a uma instância em EC2 -VPC.

```
aws ec2 describe-addresses \  
  --allocation-ids eipalloc-282d9641
```

Saída:

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

Exemplo 4: Para recuperar detalhes sobre um endereço IP elástico especificado por seu endereço IP VPC privado

O `describe-addresses` exemplo a seguir exibe detalhes sobre o endereço IP elástico associado a um determinado endereço IP privado em EC2 -VPC.

```
aws ec2 describe-addresses \  
  --filters "Name=private-ip-address,Values=10.251.50.12"
```

Exemplo 5: Para recuperar detalhes sobre endereços IP elásticos em EC2 -Classic

O `describe-addresses` exemplo a seguir exibe detalhes sobre seus endereços IP elásticos para uso no EC2 -Classic.

```
aws ec2 describe-addresses \  
  --filters "Name=domain,Values=standard"
```

Saída:


```
{
  "Addresses": [
    {
      "InstanceId": "i-1234567890abcdef0",
      "PublicIp": "203.0.110.25",
      "PublicIpv4Pool": "amazon",
      "Domain": "standard"
    }
  ]
}
```

Exemplo 6: recuperar detalhes sobre um endereço IP elástico especificado pelo endereço IP público

O `describe-addresses` exemplo a seguir exibe detalhes sobre o endereço IP elástico com o valor `203.0.110.25`, que está associado a uma instância em EC2 -Classic.

```
aws ec2 describe-addresses \
  --public-ips 203.0.110.25
```

Saída:

```
{
  "Addresses": [
    {
      "InstanceId": "i-1234567890abcdef0",
      "PublicIp": "203.0.110.25",
      "PublicIpv4Pool": "amazon",
      "Domain": "standard"
    }
  ]
}
```

- Para API obter detalhes, consulte [DescribeAddresses](#) na Referência de AWS CLI Comandos.

JavaScript

SDK para JavaScript (v3)

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
import { DescribeAddressesCommand, EC2Client } from "@aws-sdk/client-ec2";

/**
 * Describes the specified Elastic IP addresses or all of your Elastic IP
 * addresses.
 * @param {{ allocationId: string }} options
 */
export const main = async ({ allocationId }) => {
  const client = new EC2Client({});
  const command = new DescribeAddressesCommand({
    // You can omit this property to show all addresses.
    AllocationIds: [allocationId],
  });

  try {
    const { Addresses } = await client.send(command);
    const addressList = Addresses.map((address) => ` • ${address.PublicIp}`);
    console.log("Elastic IP addresses:");
    console.log(addressList.join("\n"));
  } catch (caught) {
    if (
      caught instanceof Error &&
      caught.name === "InvalidAllocationID.NotFound"
    ) {
      console.warn(`${caught.message}. Please provide a valid AllocationId.`);
    } else {
      throw caught;
    }
  }
};
```

- Para API obter detalhes, consulte [DescribeAddresses](#) em AWS SDK for JavaScript API Referência.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo descreve o endereço IP elástico especificado para instâncias em EC2 -Classic.

```
Get-EC2Address -AllocationId eipalloc-12345678
```

Saída:

```
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
```

Exemplo 2: Este exemplo descreve seus endereços IP elásticos para instâncias em VPC a. Essa sintaxe requer a PowerShell versão 3 ou posterior.

```
Get-EC2Address -Filter @{ Name="domain";Values="vpc" }
```

Exemplo 3: Este exemplo descreve o endereço IP elástico especificado para instâncias em EC2 -Classic.

```
Get-EC2Address -PublicIp 203.0.113.17
```

Saída:

```
AllocationId      :
AssociationId     :
Domain           : standard
InstanceId        : i-12345678
```

```
NetworkInterfaceId      :
NetworkInterfaceOwnerId :
PrivateIpAddress        :
PublicIp                 : 203.0.113.17
```

Exemplo 4: Este exemplo descreve seus endereços IP elásticos para instâncias em EC2 - Classic. Essa sintaxe requer a PowerShell versão 3 ou posterior.

```
Get-EC2Address -Filter @{ Name="domain";Values="standard" }
```

Exemplo 5: Este exemplo descreve todos os seus endereços IP elásticos.

```
Get-EC2Address
```

Exemplo 6: Este exemplo retorna o IP público e privado para o ID da instância fornecido no filtro

```
Get-EC2Address -Region eu-west-1 -Filter @{Name="instance-
id";Values="i-0c12d3f4f567ffb89"} | Select-Object PrivateIpAddress, PublicIp
```

Saída:

```
PrivateIpAddress PublicIp
-----
10.0.0.99         63.36.5.227
```

Exemplo 7: Este exemplo recupera todo o Elastic IPs com seu ID de alocação, ID de associação e IDs de instância

```
Get-EC2Address -Region eu-west-1 | Select-Object InstanceId, AssociationId,
AllocationId, PublicIp
```

Saída:

```
InstanceId          AssociationId          AllocationId
-----
PublicIp
-----
-----
```

```

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

```

Exemplo 8: Este exemplo busca uma lista de endereços EC2 IP que correspondem à chave de tag 'Category' com o valor 'Prod'

```
Get-EC2Address -Filter @{"Name"="tag:Category";Values="Prod"}
```

Saída:

```

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}

```

- Para API obter detalhes, consulte [DescribeAddresses](#) em Referência de AWS Tools for PowerShell cmdlet.

SAP ABAP

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
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.
```

- Para API obter detalhes, consulte [DescribeAddresses AWS SDK para SAP ABAP API](#) referência.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **DescribeAvailabilityZones** com um AWS SDK ou CLI

Os exemplos de código a seguir mostram como usar o DescribeAvailabilityZones.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação no contexto no seguinte exemplo de código:

- [Criar e gerenciar um serviço resiliente](#)

.NET

AWS SDK for .NET

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
/// <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()
{
    try
    {
        var zoneResponse = await _amazonEc2.DescribeAvailabilityZonesAsync(
            new DescribeAvailabilityZonesRequest());
        return zoneResponse.AvailabilityZones.Select(z =>
z.ZoneName).ToList();
    }
    catch (AmazonEC2Exception ec2Exception)
    {
        _logger.LogError($"An Amazon EC2 error occurred while listing
availability zones.: {ec2Exception.Message}");
        throw;
    }
    catch (Exception ex)
    {
        _logger.LogError($"An error occurred while listing availability
zones.: {ex.Message}");
        throw;
    }
}
```

- Para API obter detalhes, consulte [DescribeAvailabilityZones](#) em AWS SDK for .NET API Referência.

C++

SDK para C++

 Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
#!/ DescribeAvailabilityZones
/*!
 \param clientConfiguration: AWS client configuration.
 \return bool: Function succeeded.
*/
int AwsDoc::EC2::describeAvailabilityZones(const Aws::Client::ClientConfiguration
&clientConfiguration) {
    Aws::EC2::EC2Client ec2Client(clientConfiguration);
    Aws::EC2::Model::DescribeAvailabilityZonesRequest request;
    Aws::EC2::Model::DescribeAvailabilityZonesOutcome outcome =
    ec2Client.DescribeAvailabilityZones(request);

    if (outcome.IsSuccess()) {
        std::cout << std::left <<
            std::setw(32) << "ZoneName" <<
            std::setw(20) << "State" <<
            std::setw(32) << "Region" << std::endl;

        const auto &zones =
            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:" <<
            outcome.GetError().GetMessage() << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Para API obter detalhes, consulte [DescribeAvailabilityZones](#) em AWS SDK for C++ API Referência.

CLI

AWS CLI

Para descrever suas zonas de disponibilidade

O exemplo `describe-availability-zones` a seguir exibe detalhes das zonas de disponibilidade disponíveis para você. A resposta inclui zonas de disponibilidade somente para a região atual. Neste exemplo, ela usa a região padrão dos perfis do `us-west-2` (Oregon).

```
aws ec2 describe-availability-zones
```

Saída:

```
{
  "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"
  }
]
}

```

- Para API obter detalhes, consulte [DescribeAvailabilityZones](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo descreve as zonas de disponibilidade da região atual que estão disponíveis para você.

```
Get-EC2AvailabilityZone
```

Saída:

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

Exemplo 2: Este exemplo descreve todas as zonas de disponibilidade que estão em estado de comprometimento. A sintaxe usada neste exemplo requer a PowerShell versão 3 ou superior.

```
Get-EC2AvailabilityZone -Filter @{ Name="state";Values="impaired" }
```

Exemplo 3: Com a PowerShell versão 2, você deve usar `New-Object` para criar o filtro.

```
$filter = New-Object Amazon.EC2.Model.Filter
$filter.Name = "state"
$filter.Values = "impaired"

Get-EC2AvailabilityZone -Filter $filter
```

- Para API obter detalhes, consulte [DescribeAvailabilityZones](#) em Referência de AWS Tools for PowerShell cmdlet.

Python

SDK para Python (Boto3)

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
class AutoScalingWrapper:
    """
    Encapsulates Amazon EC2 Auto Scaling and EC2 management actions.
    """

    def __init__(
        self,
        resource_prefix: str,
        inst_type: str,
        ami_param: str,
        autoscaling_client: boto3.client,
        ec2_client: boto3.client,
        ssm_client: boto3.client,
        iam_client: boto3.client,
    ):
        """
        Initializes the AutoScaler class with the necessary parameters.

        :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
sts_client = boto3.client("sts")
self.account_id = sts_client.get_caller_identity()["Account"]

self.key_pair_name = f"{resource_prefix}-key-pair"
self.launch_template_name = f"{resource_prefix}-template-"
self.group_name = f"{resource_prefix}-group"

# Happy path
self.instance_policy_name = f"{resource_prefix}-pol"
self.instance_role_name = f"{resource_prefix}-role"
self.instance_profile_name = f"{resource_prefix}-prof"

# Failure mode
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"

def get_availability_zones(self) -> List[str]:
    """
    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"]]
        log.info(f"Retrieved {len(zones)} availability zones: {zones}.")
    except ClientError as err:
        log.error("Failed to retrieve availability zones.")
        log.error(f"Full error:\n\t{err}")
    else:
        return zones
```

- Para API obter detalhes, consulte a [DescribeAvailabilityZones](#) Referência AWS SDK do Python (Boto3). API

SAP ABAP

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
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.
```

- Para API obter detalhes, consulte [DescribeAvailabilityZones AWS SDK para SAP ABAP API](#) referência.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **DescribeBundleTasks** com um CLI

Os exemplos de código a seguir mostram como usar o `DescribeBundleTasks`.

CLI

AWS CLI

Para descrever suas tarefas do pacote

Este exemplo descreve todas as suas tarefas do pacote.

Comando:

```
aws ec2 describe-bundle-tasks
```

Saída:

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

- Para API obter detalhes, consulte [DescribeBundleTasks](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo descreve a tarefa de pacote especificada.

```
Get-EC2BundleTask -BundleId bun-12345678
```

Exemplo 2: Este exemplo descreve as tarefas do pacote cujo estado é “concluído” ou “falhado”.

```
$filter = New-Object Amazon.EC2.Model.Filter
$filter.Name = "state"
$filter.Values = @( "complete", "failed" )

Get-EC2BundleTask -Filter $filter
```

- Para API obter detalhes, consulte [DescribeBundleTasks](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **DescribeCapacityReservations** com um CLI

Os exemplos de código a seguir mostram como usar o DescribeCapacityReservations.

CLI

AWS CLI

Exemplo 1: Para descrever uma ou mais de suas reservas de capacidade

O describe-capacity-reservations exemplo a seguir exibe detalhes sobre todas as suas reservas de capacidade na AWS região atual.

```
aws ec2 describe-capacity-reservations
```

Saída:

```
{
  "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"
  }
]
}

```

Exemplo 2: Para descrever uma ou mais de suas reservas de capacidade

O `describe-capacity-reservations` exemplo a seguir exibe detalhes sobre a reserva de capacidade especificada.

```

aws ec2 describe-capacity-reservations \
  --capacity-reservation-ids cr-1234abcd56EXAMPLE

```

Saída:

```

{
  "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"
    }
]
}
```

Para obter mais informações, consulte [Visualizar uma reserva de capacidade](#) no Guia do usuário do Amazon Elastic Compute Cloud para instâncias Linux.

- Para API obter detalhes, consulte [DescribeCapacityReservations](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo descreve uma ou mais de suas reservas de capacidade para a região

```
Get-EC2CapacityReservation -Region eu-west-1
```

Saída:

```
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
```

- Para API obter detalhes, consulte [DescribeCapacityReservations](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **DescribeCustomerGateways** com um CLI

Os exemplos de código a seguir mostram como usar o `DescribeCustomerGateways`.

CLI

AWS CLI

Para descrever os gateways de seus clientes

Este exemplo descreve os gateways do seu cliente.

Comando:

```
aws ec2 describe-customer-gateways
```

Saída:

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

Para descrever um gateway de cliente específico

Este exemplo descreve o gateway do cliente especificado.

Comando:

```
aws ec2 describe-customer-gateways --customer-gateway-ids cgw-0e11f167
```

Saída:

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

- Para API obter detalhes, consulte [DescribeCustomerGateways](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo descreve o gateway do cliente especificado.

```
Get-EC2CustomerGateway -CustomerGatewayId cgw-1a2b3c4d
```

Saída:

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

Exemplo 2: Este exemplo descreve qualquer gateway de cliente cujo estado esteja pendente ou disponível.

```
$filter = New-Object Amazon.EC2.Model.Filter
$filter.Name = "state"
$filter.Values = @( "pending", "available" )

Get-EC2CustomerGateway -Filter $filter
```

Exemplo 3: Este exemplo descreve todos os gateways de seus clientes.

```
Get-EC2CustomerGateway
```

- Para API obter detalhes, consulte [DescribeCustomerGateways](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **DescribeDhcpOptions** com um CLI

Os exemplos de código a seguir mostram como usar o DescribeDhcpOptions.

CLI

AWS CLI

Exemplo 1: Para descrever suas DHCP opções

O describe-dhcp-options exemplo a seguir recupera detalhes sobre suas DHCP opções.

```
aws ec2 describe-dhcp-options
```

Saída:

```
{
  "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"
  }
]
}

```

Para obter mais informações, consulte [Trabalhando com conjuntos de DHCP opções](#) no Guia AWS VPC do usuário.

Exemplo 2: Para descrever suas DHCP opções e filtrar a saída

O `describe-dhcp-options` exemplo a seguir descreve suas DHCP opções e usa um filtro para retornar somente DHCP as opções que você tem `example.com` para o servidor de nomes de domínio. O exemplo usa o `--query` parâmetro para exibir somente as informações de configuração e o ID na saída.

```

aws ec2 describe-dhcp-options \
  --filters Name=key,Values=domain-name-servers Name=value,Values=example.com \
  --query "DhcpOptions[*].[DhcpConfigurations,DhcpOptionsId]"

```

Saída:

```

[
  [
    [
      {
        "Key": "domain-name",
        "Values": [
          {
            "Value": "example.com"
          }
        ]
      },
      {
        "Key": "domain-name-servers",
        "Values": [
          {
            "Value": "172.16.16.16"
          }
        ]
      }
    ]
  ]
]

```

```
    ],
    "dopt-001122334455667ab"
  ]
]
```

Para obter mais informações, consulte [Trabalhando com conjuntos de DHCP opções](#) no Guia AWS VPC do usuário.

- Para API obter detalhes, consulte [DescribeDhcpOptions](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo lista seus conjuntos DHCP de opções.

```
Get-EC2DhcpOption
```

Saída:

DhcpConfigurations	DhcpOptionsId	Tag
-----	-----	---
{domain-name, domain-name-servers}	dopt-1a2b3c4d	{}
{domain-name, domain-name-servers}	dopt-2a3b4c5d	{}
{domain-name-servers}	dopt-3a4b5c6d	{}

Exemplo 2: Este exemplo obtém detalhes de configuração para o conjunto DHCP de opções especificado.

```
(Get-EC2DhcpOption -DhcpOptionsId dopt-1a2b3c4d).DhcpConfigurations
```

Saída:

Key	Values
---	-----
domain-name	{abc.local}
domain-name-servers	{10.0.0.101, 10.0.0.102}

- Para API obter detalhes, consulte [DescribeDhcpOptions](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use `DescribeFlowLogs` com um CLI

Os exemplos de código a seguir mostram como usar o `DescribeFlowLogs`.

CLI

AWS CLI

Exemplo 1: Para descrever todos os seus registros de fluxo

O `describe-flow-logs` exemplo a seguir exibe detalhes de todos os seus registros de fluxo.

```
aws ec2 describe-flow-logs
```

Saída:

```
{
  "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}"
    }
]
}

```

Exemplo 2: Para descrever um subconjunto dos seus registros de fluxo

O `describe-flow-logs` exemplo a seguir usa um filtro para exibir detalhes somente dos registros de fluxo que estão no grupo de registros especificado no Amazon CloudWatch Logs.

```

aws ec2 describe-flow-logs \
  --filter "Name=log-group-name,Values=MyFlowLogs"

```

- Para API obter detalhes, consulte [DescribeFlowLogs](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo descreve um ou mais registros de fluxo com o tipo de destino de log 's3'

```

Get-EC2FlowLog -Filter @{"Name="log-destination-type";Values="s3"}

```

Saída:

```

CreationTime           : 2/25/2019 9:07:36 PM
DeliverLogsErrorMessage :
DeliverLogsPermissionArn :
DeliverLogsStatus      : SUCCESS
FlowLogId              : f1-01b2e3d45f67f8901
FlowLogStatus          : ACTIVE
LogDestination         : arn:aws:s3::my-bucket-dd-tata

```

```
LogDestinationType      : s3
LogGroupName            :
ResourceId               : eni-01d2dda3456b7e890
TrafficType             : ALL
```

- Para API obter detalhes, consulte [DescribeFlowLogs](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use `DescribeHostReservationOfferings` com um CLI

Os exemplos de código a seguir mostram como usar o `DescribeHostReservationOfferings`.

CLI

AWS CLI

Para descrever as ofertas de reserva de anfitriões dedicados

Este exemplo descreve as reservas de host dedicado para a família de instâncias M4 que estão disponíveis para compra.

Comando:

```
aws ec2 describe-host-reservation-offerings --filter Name=instance-  
family,Values=m4
```

Saída:

```
{
  "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
}
]
```

- Para API obter detalhes, consulte [DescribeHostReservationOfferings](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo descreve as reservas de host dedicado que estão disponíveis para compra para o determinado filtro 'instance-family', onde está PaymentOption " NoUpfront

```
Get-EC2HostReservationOffering -Filter @{Name="instance-family";Values="m4"} |  
Where-Object PaymentOption -eq NoUpfront
```

Saída:

```
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
```

- Para API obter detalhes, consulte [DescribeHostReservationOfferings](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **DescribeHosts** com um CLI

Os exemplos de código a seguir mostram como usar o DescribeHosts.

CLI

AWS CLI

Para ver detalhes sobre hosts dedicados

O describe-hosts exemplo a seguir exibe detalhes dos hosts available dedicados em sua AWS conta.

```
aws ec2 describe-hosts --filter "Name=state,Values=available"
```

Saída:

```
{
  "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"
}
]
}

```

Para obter mais informações, consulte [Visualização de hosts dedicados](#) no Guia do usuário do Amazon Elastic Compute Cloud para instâncias Linux.

- Para API obter detalhes, consulte [DescribeHosts](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo retorna os detalhes do EC2 host

```
Get-EC2Host
```

Saída:

```

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                : {}

```

Exemplo 2: Este exemplo consulta o host AvailableInstanceCapacity h-01e23f4cd567899f1

```
Get-EC2Host -HostId h-01e23f4cd567899f1 | Select-Object -ExpandProperty
AvailableCapacity | Select-Object -expand AvailableInstanceCapacity
```

Saída:

```

AvailableCapacity InstanceType TotalCapacity
-----
11                m4.xlarge    11

```

- Para API obter detalhes, consulte [DescribeHosts](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use `DescribeIamInstanceProfileAssociations` com um AWS SDK ou CLI

Os exemplos de código a seguir mostram como usar o `DescribeIamInstanceProfileAssociations`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação no contexto no seguinte exemplo de código:

- [Criar e gerenciar um serviço resiliente](#)

.NET**AWS SDK for .NET****Note**

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```

/// <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)

```



```
{
    try
    {
        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];
    }
    catch (AmazonEC2Exception ec2Exception)
    {
        if (ec2Exception.ErrorCode == "InvalidInstanceID.NotFound")
        {
            _logger.LogError(ec2Exception, $"Instance {instanceId} not
found");
        }

        throw;
    }
    catch (Exception ex)
    {
        _logger.LogError(ex, $"An error occurred while creating the
template.: {ex.Message}");
        throw;
    }
}
```

- Para API obter detalhes, consulte [DescribeIamInstanceProfileAssociation](#) em AWS SDK for .NET API Referência.

CLI

AWS CLI

Para descrever associações de perfis de IAM instância

Este exemplo descreve todas as associações do perfil da sua IAM instância.

Comando:

```
aws ec2 describe-iam-instance-profile-associations
```

Saída:

```
{
  "IamInstanceProfileAssociations": [
    {
      "InstanceId": "i-09eb09efa73ec1dee",
      "State": "associated",
      "AssociationId": "iip-assoc-0db249b1f25fa24b8",
      "IamInstanceProfile": {
        "Id": "AIPAJVQN4F5WVLGCJDRGM",
        "Arn": "arn:aws:iam::123456789012:instance-profile/admin-role"
      }
    },
    {
      "InstanceId": "i-0402909a2f4dfffd14",
      "State": "associating",
      "AssociationId": "iip-assoc-0d1ec06278d29f44a",
      "IamInstanceProfile": {
        "Id": "AGJAJVQN4F5WVLGCJABCM",
        "Arn": "arn:aws:iam::123456789012:instance-profile/user1-role"
      }
    }
  ]
}
```

- Para API obter detalhes, consulte [Descreva as Associações de Perfil de Instância IAM](#) na Referência de AWS CLI Comandos.

JavaScript

SDK para JavaScript (v3)

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
const ec2Client = new EC2Client({});
const { IamInstanceProfileAssociations } = await ec2Client.send(
  new DescribeIamInstanceProfileAssociationsCommand({
    Filters: [
      { Name: "instance-id", Values: [state.targetInstance.InstanceId] },
    ],
  }),
);
```

- Para API obter detalhes, consulte [DescrevaIamInstanceProfileAssociation](#) em AWS SDK for JavaScript API Referência.

Python

SDK para Python (Boto3)

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
class AutoScalingWrapper:
    """
    Encapsulates Amazon EC2 Auto Scaling and EC2 management actions.
    """

    def __init__(
        self,
        resource_prefix: str,
        inst_type: str,
        ami_param: str,
        autoscaling_client: boto3.client,
        ec2_client: boto3.client,
        ssm_client: boto3.client,
        iam_client: boto3.client,
    ):
        """
        Initializes the AutoScaler class with the necessary parameters.
```

```

        :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
        sts_client = boto3.client("sts")
        self.account_id = sts_client.get_caller_identity()["Account"]

        self.key_pair_name = f"{resource_prefix}-key-pair"
        self.launch_template_name = f"{resource_prefix}-template-"
        self.group_name = f"{resource_prefix}-group"

        # Happy path
        self.instance_policy_name = f"{resource_prefix}-pol"
        self.instance_role_name = f"{resource_prefix}-role"
        self.instance_profile_name = f"{resource_prefix}-prof"

        # Failure mode
        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"

    def get_instance_profile(self, instance_id: str) -> Dict[str, Any]:
        """
        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]}]
    )
    if not response["IamInstanceProfileAssociations"]:
        log.info(f"No instance profile found for instance
{instance_id}.")
        profile_data = response["IamInstanceProfileAssociations"][0]
        log.info(f"Retrieved instance profile for instance {instance_id}.")
        return profile_data
    except ClientError as err:
        log.error(
            f"Failed to retrieve instance profile for instance
{instance_id}."
        )
        error_code = err.response["Error"]["Code"]
        if error_code == "InvalidInstanceID.NotFound":
            log.error(f"The instance ID '{instance_id}' does not exist.")
            log.error(f"Full error:\n\t{err}")

```

- Para API obter detalhes, consulte a [Descreva as Associações de Perfil de Instância](#) Referência AWS SDK do Python (Boto3). API

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **DescribeIdFormat** com um CLI

Os exemplos de código a seguir mostram como usar o `DescribeIdFormat`.

CLI

AWS CLI

Exemplo 1: Para descrever o formato de ID de um recurso

O `describe-id-format` exemplo a seguir descreve o formato de ID para grupos de segurança.

```
aws ec2 describe-id-format \
    --resource security-group
```

No exemplo de saída a seguir, o `Deadline` valor indica que o prazo para esse tipo de recurso mudar permanentemente do formato de ID curto para o formato de ID longo expirou às 00:00 de 15 de UTC agosto de 2018.

```
{
  "Statuses": [
    {
      "Deadline": "2018-08-15T00:00:00.000Z",
      "Resource": "security-group",
      "UseLongIds": true
    }
  ]
}
```

Exemplo 2: Para descrever o formato de ID de todos os recursos

O `describe-id-format` exemplo a seguir descreve o formato de ID para todos os tipos de recursos. Todos os tipos de recursos compatíveis com o formato de ID curto foram alterados para o formato de ID longo.

```
aws ec2 describe-id-format
```

- Para API obter detalhes, consulte [DescribeIdFormat](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo descreve o formato de ID para o tipo de recurso especificado.

```
Get-EC2IdFormat -Resource instance
```

Saída:

Resource	UseLongIds
-----	-----
instance	False

Exemplo 2: Este exemplo descreve os formatos de ID para todos os tipos de recursos que oferecem suporte por mais tempoIDs.

```
Get-EC2IdFormat
```

Saída:

```
Resource      UseLongIds
-----
reservation   False
instance      False
```

- Para API obter detalhes, consulte [DescribeIdFormat](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use `DescribeIdentityIdFormat` com um CLI

Os exemplos de código a seguir mostram como usar o `DescribeIdentityIdFormat`.

CLI

AWS CLI

Para descrever o formato de ID de uma IAM função

O `describe-identity-id-format` exemplo a seguir descreve o formato de ID recebido pelas instâncias criadas pela IAM função `EC2Role` em sua AWS conta.

```
aws ec2 describe-identity-id-format \
  --principal-arn arn:aws:iam::123456789012:role/my-iam-role \
  --resource instance
```

O resultado a seguir indica que as instâncias criadas por essa função recebem IDs em formato de ID longo.

```
{
  "Statuses": [
    {
      "Deadline": "2016-12-15T00:00:00Z",
```

```

        "Resource": "instance",
        "UseLongIds": true
    }
]
}

```

Para descrever o formato de ID de um IAM usuário

O `describe-identity-id-format` exemplo a seguir descreve o formato de ID recebido pelos instantâneos criados pelo IAM usuário `AdminUser` em sua AWS conta.

```

aws ec2 describe-identity-id-format \
  --principal-arn arn:aws:iam::123456789012:user/AdminUser \
  --resource snapshot

```

A saída indica que os instantâneos criados por esse usuário são recebidos IDs no formato de ID longo.

```

{
  "Statuses": [
    {
      "Deadline": "2016-12-15T00:00:00Z",
      "Resource": "snapshot",
      "UseLongIds": true
    }
  ]
}

```

- Para API obter detalhes, consulte [DescribeIdentityIdFormat](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo retorna o formato de ID do recurso 'imagem' para a função fornecida

```

Get-EC2IdentityIdFormat -PrincipalArn arn:aws:iam::123456789511:role/JDBC -
Resource image

```

Saída:


```

Deadline                Resource UseLongIds
-----                -
8/2/2018 11:30:00 PM image    True

```

- Para API obter detalhes, consulte [DescribeIdentityIdFormat](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use `DescribeImageAttribute` com um CLI

Os exemplos de código a seguir mostram como usar o `DescribeImageAttribute`.

CLI

AWS CLI

Para descrever as permissões de lançamento de um AMI

Este exemplo descreve as permissões de lançamento para o especificado AMI.

Comando:

```
aws ec2 describe-image-attribute --image-id ami-5731123e --
attribute LaunchPermission
```

Saída:

```
{
  "LaunchPermissions": [
    {
      "UserId": "123456789012"
    }
  ],
  "ImageId": "ami-5731123e",
}
```

Para descrever os códigos de produto para um AMI

Este exemplo descreve os códigos de produto para o especificadoAMI. Observe que isso não AMI tem códigos de produto.

Comando:

```
aws ec2 describe-image-attribute --image-id ami-5731123e --attribute productCodes
```

Saída:

```
{  
  "ProductCodes": [],  
  "ImageId": "ami-5731123e",  
}
```

- Para API obter detalhes, consulte [DescribeImageAttribute](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo obtém a descrição do especificadoAMI.

```
Get-EC2ImageAttribute -ImageId ami-12345678 -Attribute description
```

Saída:

```
BlockDeviceMappings : {}  
Description          : My image description  
ImageId              : ami-12345678  
KernelId             :  
LaunchPermissions    : {}  
ProductCodes         : {}  
RamdiskId            :  
SriovNetSupport      :
```

Exemplo 2: Este exemplo obtém as permissões de lançamento para o especificadoAMI.

```
Get-EC2ImageAttribute -ImageId ami-12345678 -Attribute launchPermission
```

Saída:

```
BlockDeviceMappings : {}
Description          :
ImageId              : ami-12345678
KernelId             :
LaunchPermissions    : {all}
ProductCodes         : {}
RamdiskId            :
SriovNetSupport      :
```

Exemplo 3: Este exemplo testa se a rede avançada está habilitada.

```
Get-EC2ImageAttribute -ImageId ami-12345678 -Attribute sriovNetSupport
```

Saída:

```
BlockDeviceMappings : {}
Description          :
ImageId              : ami-12345678
KernelId             :
LaunchPermissions    : {}
ProductCodes         : {}
RamdiskId            :
SriovNetSupport      : simple
```

- Para API obter detalhes, consulte [DescribeImageAttribute](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **DescribeImages** com um AWS SDK ou CLI

Os exemplos de código a seguir mostram como usar o DescribeImages.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação no contexto no seguinte exemplo de código:

- [Aprender os conceitos básicos](#)

Bash

AWS CLI com script Bash

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
#####
# 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 getopt 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 getopt "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
}

```

As funções utilitárias usadas neste exemplo.

```

#####
# 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
}

```

- Para API obter detalhes, consulte [DescrevaImagens](#) na Referência de AWS CLI Comandos.

CLI

AWS CLI

Exemplo 1: Para descrever um AMI

O `describe-images` exemplo a seguir descreve o especificado AMI na região especificada.

```
aws ec2 describe-images \  
  --region us-east-1 \  
  --image-ids ami-1234567890EXAMPLE
```

Saída:

```
{  
  "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"
    }
]
}

```

Para obter mais informações, consulte [Amazon Machine Images \(AMI\)](#) no Guia EC2 do usuário da Amazon.

Exemplo 2: Para descrever AMIs com base em filtros

O `describe-images` exemplo a seguir descreve o Windows AMIs fornecido pela Amazon que é apoiado pela AmazonEBS.

```

aws ec2 describe-images \
  --owners amazon \
  --filters "Name=platform,Values=windows" "Name=root-device-type,Values=ebs"

```

Para obter um exemplo da saída de `describe-images`, consulte o Exemplo 1.

Para obter exemplos adicionais de uso de filtros, consulte [Listar e filtrar seus recursos](#) no Guia do EC2 usuário da Amazon.

Exemplo 3: Para descrever AMIs com base em tags

O `describe-images` exemplo a seguir descreve todos os AMIs que têm a `tagType=Custom`. O exemplo usa o `--query` parâmetro para exibir somente AMI IDs o.

```

aws ec2 describe-images \
  --filters "Name=tag:Type,Values=Custom" \
  --query 'Images[*].[ImageId]' \
  --output text

```

Saída:

```

ami-1234567890EXAMPLE
ami-0abcdef1234567890

```


Para obter mais exemplos de uso de filtros de tags, consulte Como [trabalhar com tags](#) no Guia EC2 do usuário da Amazon.

- Para API obter detalhes, consulte [DescribeImages](#) na Referência de AWS CLI Comandos.

JavaScript

SDK para JavaScript (v3)

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
import { EC2Client, paginateDescribeImages } from "@aws-sdk/client-ec2";

/**
 * Describes the specified images (AMIs, AKIs, and ARIs) available to you or all
 * of the images available to you.
 * @param {{ architecture: string, pageSize: number }} options
 */
export const main = async ({ architecture, pageSize }) => {
  pageSize = Number.parseInt(pageSize);
  const client = new EC2Client({});

  // 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 },
    {
      // 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: [architecture] }],
    },
  );
};
```

```
/**
 * @type {import('@aws-sdk/client-ec2').Image[]}
 */
const images = [];
let recordsScanned = 0;

try {
  for await (const page of paginator) {
    recordsScanned += pageSize;
    if (page.Images.length) {
      images.push(...page.Images);
      break;
    }
    console.log(
      `No matching image found yet. Searched ${recordsScanned} records.`
    );
  }

  if (images.length) {
    console.log(
      `Found ${images.length} images:\n\n${images.map((image) =>
image.Name).join("\n")}\n`,
    );
  } else {
    console.log(
      `No matching images found. Searched ${recordsScanned} records.\n`,
    );
  }

  return images;
} catch (caught) {
  if (caught instanceof Error && caught.name === "InvalidParameterValue") {
    console.warn(`${caught.message}`);
    return [];
  }
  throw caught;
}
};
```

- Para API obter detalhes, consulte [DescreverImagem](#) em AWS SDK for JavaScript API Referência.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo descreve o especificadoAMI.

```
Get-EC2Image -ImageId ami-12345678
```

Saída:

```
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
```

Exemplo 2: Este exemplo descreve o AMIs que você possui.

```
Get-EC2Image -owner self
```

Exemplo 3: Este exemplo descreve o público AMIs que executa o Microsoft Windows Server.

```
Get-EC2Image -Filter @{ Name="platform"; Values="windows" }
```

Exemplo 4: Este exemplo descreve todos os públicos AMIs na região 'us-west-2'.

```
Get-EC2Image -Region us-west-2
```

- Para API obter detalhes, consulte [DescribeImages](#) em Referência de AWS Tools for PowerShell cmdlet.

Python

SDK para Python (Boto3)

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
class EC2InstanceWrapper:
    """Encapsulates Amazon Elastic Compute Cloud (Amazon EC2) instance actions
    using the client interface."""

    def __init__(
        self, ec2_client: Any, instances: Optional[List[Dict[str, Any]]] = None
    ) -> None:
        """
        Initializes the EC2InstanceWrapper with an EC2 client and optional
        instances.

        :param ec2_client: A Boto3 Amazon EC2 client. This client provides low-
        level
                           access to AWS EC2 services.
        :param instances: A list of dictionaries representing Boto3 Instance
        objects. These are high-level objects that
                           wrap instance actions.
        """
        self.ec2_client = ec2_client
        self.instances = instances or []

    @classmethod
    def from_client(cls) -> "EC2InstanceWrapper":
        """
```

```
Creates an EC2InstanceWrapper instance with a default EC2 client.

:return: An instance of EC2InstanceWrapper initialized with the default
EC2 client.
"""
ec2_client = boto3.client("ec2")
return cls(ec2_client)

def get_images(self, image_ids: List[str]) -> List[Dict[str, Any]]:
    """
    Gets information about Amazon Machine Images (AMIs) from a list of AMI
    IDs.

    :param image_ids: The list of AMI IDs to look up.
    :return: A list of dictionaries representing the requested AMIs.
    """
    try:
        response = self.ec2_client.describe_images(ImageIds=image_ids)
        images = response["Images"]
    except ClientError as err:
        logger.error(f"Failed to stop AMI(s): {','.join(map(str,
image_ids))}")
        error_code = err.response["Error"]["Code"]
        if error_code == "InvalidAMIID.NotFound":
            logger.error("One or more of the AMI IDs does not exist.")
            raise
    return images
```

- Para API obter detalhes, consulte a [DescribeImages](#) Referência AWS SDK do Python (Boto3). API

Rust

SDK para Rust

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
pub async fn list_images(&self, ids: Vec<Parameter>) -> Result<Vec<Image>,
EC2Error> {
    let image_ids = ids.into_iter().filter_map(|p| p.value).collect();
    let output = self
        .client
        .describe_images()
        .set_image_ids(Some(image_ids))
        .send()
        .await?;

    let images = output.images.unwrap_or_default();
    if images.is_empty() {
        Err(EC2Error::new("No images for selected AMIs"))
    } else {
        Ok(images)
    }
}
```

Usando a função `list_images` com limite SSM de acordo com seu ambiente. Para obter mais detalhes SSM, consulte <https://docs.aws.amazon.com/systems-manager/latest/userguide/example-GetParameters-ssm-section.html>.

```
async fn find_image(&mut self) -> Result<ScenarioImage, EC2Error> {
    let params: Vec<Parameter> = self
        .ssm
        .list_path("/aws/service/ami-amazon-linux-latest")
        .await
        .map_err(|e| e.add_message("Could not find parameters for available
images"))?
        .into_iter()
```

```

        .filter(|param| param.name().is_some_and(|name|
name.contains("amzn2")))
        .collect();
    let amzn2_images: Vec<ScenarioImage> = self
        .ec2
        .list_images(params)
        .await
        .map_err(|e| e.add_message("Could not find images"))?
        .into_iter()
        .map(ScenarioImage::from)
        .collect();
    println!("We will now create an instance from an Amazon Linux 2 AMI");
    let ami = self.util.select_scenario_image(amzn2_images)?;
    Ok(ami)
}

```

- Para API obter detalhes, consulte [DescrevaImagens](#) a AWS SDKAPIreferência do Rust.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use `DescribeImportImageTasks` com um CLI

Os exemplos de código a seguir mostram como usar o `DescribeImportImageTasks`.

CLI

AWS CLI

Para monitorar uma tarefa de importação de imagem

O `describe-import-image-tasks` exemplo a seguir verifica o status da tarefa de importação de imagem especificada.

```
aws ec2 describe-import-image-tasks \
  --import-task-ids import-ami-1234567890abcdef0
```

Saída para uma tarefa de importação de imagem que está em andamento.

```
{
```

```

    "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"
      }
    ]
  }
}

```

Saída para uma tarefa de importação de imagem concluída. O ID do resultado AMI é fornecido por `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"
    }
  ]
}

```



```
]
}
```

- Para API obter detalhes, consulte [DescribeImportImageTasks](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo descreve a tarefa de importação de imagem especificada.

```
Get-EC2ImportImageTask -ImportTaskId import-ami-hgfedcba
```

Saída:

```
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     :
```

Exemplo 2: Este exemplo descreve todas as suas tarefas de importação de imagens.

```
Get-EC2ImportImageTask
```

Saída:

```
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 :
```

- Para API obter detalhes, consulte [DescribeImportImageTask](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **DescribeImportSnapshotTasks** com um CLI

Os exemplos de código a seguir mostram como usar o `DescribeImportSnapshotTasks`.

CLI

AWS CLI

Para monitorar uma tarefa de importação de instantâneo

O `describe-import-snapshot-tasks` exemplo a seguir verifica o status da tarefa de importação instantânea especificada.

```
aws ec2 describe-import-snapshot-tasks \
  --import-task-ids import-snap-1234567890abcdef0
```

Saída para uma tarefa de importação de instantâneo que está em andamento:

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

Saída para uma tarefa de importação de instantâneo concluída. O ID do instantâneo resultante é fornecido por `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"
        }
      }
    }
  ]
}
```

```
}

```

- Para API obter detalhes, consulte [DescribeImportSnapshotTasks](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo descreve a tarefa de importação de snapshot especificada.

```
Get-EC2ImportSnapshotTask -ImportTaskId import-snap-abcdefgh
```

Saída:

Description	ImportTaskId	SnapshotTaskDetail
-----	-----	-----
Disk Image Import 1	import-snap-abcdefgh	Amazon.EC2.Model.SnapshotTaskDetail

Exemplo 2: Este exemplo descreve todas as suas tarefas de importação de instantâneos.

```
Get-EC2ImportSnapshotTask
```

Saída:

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

- Para API obter detalhes, consulte [DescribeImportSnapshotTasks](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use `DescribeInstanceAttribute` com um CLI

Os exemplos de código a seguir mostram como usar o `DescribeInstanceAttribute`.

CLI

AWS CLI

Para descrever o tipo de instância

Este exemplo descreve o tipo de instância da instância especificada.

Comando:

```
aws ec2 describe-instance-attribute --instance-id i-1234567890abcdef0 --  
attribute instanceType
```

Saída:

```
{  
  "InstanceId": "i-1234567890abcdef0"  
  "InstanceType": {  
    "Value": "t1.micro"  
  }  
}
```

Para descrever o `disableApiTermination` atributo

Este exemplo descreve o `disableApiTermination` atributo da instância especificada.

Comando:

```
aws ec2 describe-instance-attribute --instance-id i-1234567890abcdef0 --  
attribute disableApiTermination
```

Saída:

```
{
```

```
"InstanceId": "i-1234567890abcdef0"
  "DisableApiTermination": {
    "Value": "false"
  }
}
```

Para descrever o mapeamento de dispositivos de blocos para uma instância

Este exemplo descreve o `blockDeviceMapping` atributo da instância especificada.

Comando:

```
aws ec2 describe-instance-attribute --instance-id i-1234567890abcdef0 --
attribute blockDeviceMapping
```

Saída:

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

- Para API obter detalhes, consulte [DescribeInstanceAttribute](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Esse exemplo descreve o tipo de instância da instância especificada.

```
Get-EC2InstanceAttribute -InstanceId i-12345678 -Attribute instanceType
```

Saída:

```
InstanceType           : t2.micro
```

Exemplo 2: Este exemplo descreve se a rede avançada está habilitada para a instância especificada.

```
Get-EC2InstanceAttribute -InstanceId i-12345678 -Attribute sriovNetSupport
```

Saída:

```
SriovNetSupport        : simple
```

Exemplo 3: Este exemplo descreve os grupos de segurança da instância especificada.

```
(Get-EC2InstanceAttribute -InstanceId i-12345678 -Attribute groupSet).Groups
```

Saída:

```
GroupId  
-----  
sg-12345678  
sg-45678901
```

Exemplo 4: Este exemplo descreve se a EBS otimização está habilitada para a instância especificada.

```
Get-EC2InstanceAttribute -InstanceId i-12345678 -Attribute ebsOptimized
```

Saída:

```
EbsOptimized : False
```

Exemplo 5: Esse exemplo descreve o atributo `disableApiTermination` da instância especificada.

```
Get-EC2InstanceAttribute -InstanceId i-12345678 -Attribute disableApiTermination
```

Saída:

```
DisableApiTermination : False
```

Exemplo 6: Esse exemplo descreve o atributo “`instanceInitiatedShutdownComportamento`” da instância especificada.

```
Get-EC2InstanceAttribute -InstanceId i-12345678 -Attribute  
instanceInitiatedShutdownBehavior
```

Saída:

```
InstanceInitiatedShutdownBehavior : stop
```

- Para API obter detalhes, consulte [DescribeInstanceAttribute](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use `DescribeInstanceStatus` com um AWS SDK ou CLI

Os exemplos de código a seguir mostram como usar o `DescribeInstanceStatus`.

CLI

AWS CLI

Para descrever o status de uma instância

O exemplo `describe-instance-status` a seguir descreve o status atual da instância especificada.

```
aws ec2 describe-instance-status \  
  --instance-ids i-1234567890abcdef0
```

Saída:

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

Para obter mais informações, consulte [Monitore o status de suas instâncias](#) no Guia EC2 do usuário da Amazon.

- Para API obter detalhes, consulte [DescribeInstanceStatus](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Esse exemplo descreve o status da instância especificada.

```
Get-EC2InstanceStatus -InstanceId i-12345678
```

Saída:

```
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
```

Saída:

```
Code    Name
----    -
16     running
```

```
$status.Status
```

Saída:

```
Details      Status
-----      -
{reachability} ok
```

```
$status.SystemStatus
```

Saída:

```

Details          Status
-----          -
{reachability}  ok

```

- Para API obter detalhes, consulte [DescribeInstanceStatus](#) em Referência de AWS Tools for PowerShell cmdlet.

Rust**SDK para Rust****Note**

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```

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(())
}

```

- Para API obter detalhes, consulte [DescribeInstanceStatus](#) a AWS SDK API referência do Rust.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **DescribeInstanceTypes** com um AWS SDK ou CLI

Os exemplos de código a seguir mostram como usar o `DescribeInstanceTypes`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação no contexto no seguinte exemplo de código:

- [Aprender os conceitos básicos](#)

.NET

AWS SDK for .NET

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
/// <summary>
```

```
/// Describe the instance types available.
/// </summary>
/// <returns>A list of instance type information.</returns>
public async Task<List<InstanceTypeInfo>>
DescribeInstanceTypes(ArchitectureValues architecture)
{
    try
    {
        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;
    }
    catch (AmazonEC2Exception ec2Exception)
    {
        if (ec2Exception.ErrorCode == "InvalidParameterValue")
        {
            _logger.LogError(
                $"Parameters are invalid. Ensure architecture and size
strings conform to DescribeInstanceTypes API reference.");
        }

        throw;
    }
    catch (Exception ex)
    {
        Console.WriteLine($"Couldn't delete the security group because:
{ex.Message}");
    }
}
```

```

        throw;
    }
}

```

- Para API obter detalhes, consulte [DescribeInstanceTypes](#) em AWS SDK for .NET API Referência.

Bash

AWS CLI com script Bash

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```

#####
# 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
```



```
}

```

As funções utilitárias usadas neste exemplo.

```
#####
# 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
}
```

- Para API obter detalhes, consulte [DescribeInstanceTypes](#) na Referência de AWS CLI Comandos.

CLI

AWS CLI

Exemplo 1: descrever um tipo de instância

O exemplo `describe-instance-types` a seguir exibe os detalhes do tipo da instância especificado.

```
aws ec2 describe-instance-types \
  --instance-types t2.micro
```

Saída:

```
{
  "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
}
]
```

Para obter mais informações, consulte [Tipos de instância](#) no Guia do usuário do Amazon Elastic Compute Cloud para instâncias Linux.

Exemplo 2: filtrar os tipos de instância disponíveis

Você pode especificar um filtro para definir o escopo dos resultados para os tipos de instância que tenham uma característica específica. O exemplo `describe-instance-types` a seguir lista os tipos de instância compatíveis com hibernação.

```
aws ec2 describe-instance-types \  
  --filters Name=hibernation-supported,Values=true --query  
  'InstanceTypes[*].InstanceType'
```

Saída:


```
[  
  "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"  
]
```

Para obter mais informações, consulte [Tipos de instância](#) no Guia do usuário do Amazon Elastic Compute Cloud para instâncias Linux.

- Para API obter detalhes, consulte [DescribeInstanceTypes](#) na Referência de AWS CLI Comandos.

Java

SDK para Java 2.x

 Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
/**
 * Asynchronously retrieves the instance types available in the current AWS
 region.
 * <p>
 * This method uses the AWS SDK's asynchronous API to fetch the available
 instance types
 * and then processes the response. It logs the memory information, network
 information,
 * and instance type for each instance type returned. Additionally, it
 returns a
 * {@link CompletableFuture} that resolves to the instance type string for
 the "t2.2xlarge"
 * instance type, if it is found in the response. If the "t2.2xlarge"
 instance type is not
 * found, an empty string is returned.
 * </p>
 *
 * @return a {@link CompletableFuture} that resolves to the instance type
 string for the
 * "t2.2xlarge" instance type, or an empty string if the instance type is not
 found
 */
public CompletableFuture<String> getInstanceTypesAsync() {
    DescribeInstanceTypesRequest typesRequest =
DescribeInstanceTypesRequest.builder()
        .maxResults(10)
        .build();

    CompletableFuture<DescribeInstanceTypesResponse> response =
getAsyncClient().describeInstanceTypes(typesRequest);
    response.whenComplete((resp, ex) -> {
        if (resp != null) {
```

```
        List<InstanceTypeInfo> instanceTypes = resp.instanceTypes();
        for (InstanceTypeInfo type : instanceTypes) {
            logger.info("The memory information of this type is " +
type.memoryInfo().sizeInMiB());
            logger.info("Network information is " +
type.networkInfo().toString());
            logger.info("Instance type is " +
type.instanceType().toString());
        }
    } else {
        throw (RuntimeException) ex;
    }
});

return response.thenApply(resp -> {
    for (InstanceTypeInfo type : resp.instanceTypes()) {
        String instanceType = type.instanceType().toString();
        if (instanceType.equals("t2.2xlarge")) {
            return instanceType;
        }
    }
    return "";
});
}
```

- Para API obter detalhes, consulte [DescribeInstanceTypes](#) em AWS SDK for Java 2.x API Referência.

JavaScript

SDK para JavaScript (v3)

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
import { EC2Client, paginateDescribeInstanceTypes } from "@aws-sdk/client-ec2";
```

```
/**
 * Describes the specified instance types. By default, all instance types for the
 * current Region are described. Alternatively, you can filter the results.
 * @param {{ pageSize: string, supportedArch: string[], freeTier: boolean }}
options
 */
export const main = async ({ pageSize, supportedArch, freeTier }) => {
  pageSize = Number.parseInt(pageSize);
  const client = new EC2Client({});

  // 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 },
    {
      Filters: [
        {
          Name: "processor-info.supported-architecture",
          Values: supportedArch,
        },
        { Name: "free-tier-eligible", Values: [freeTier ? "true" : "false"] },
      ],
    },
  );

  try {
    /**
     * @type {import('@aws-sdk/client-ec2').InstanceTypeInfo[]}
     */
    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(
```

```

    `Memory size in MiB for matching instance types:\n\n
    ${instanceTypes.map((it) => `${it.InstanceType}: ${it.MemoryInfo.SizeInMiB}
    MiB`).join("\n")}` ,
    );
  } catch (caught) {
    if (caught instanceof Error && caught.name === "InvalidParameterValue") {
      console.warn(`${caught.message}`);
      return [];
    }
    throw caught;
  }
};

```

- Para API obter detalhes, consulte [DescribeInstanceTypes](#) em AWS SDK for JavaScript API Referência.

Kotlin

SDK para Kotlin

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```

// 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
    }
}

```

- Para API obter detalhes, consulte a [DescribeInstanceTypes](#) referência AWS SDK do Kotlin API.

Python

SDK para Python (Boto3)

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```

class EC2InstanceWrapper:
    """Encapsulates Amazon Elastic Compute Cloud (Amazon EC2) instance actions
    using the client interface."""

    def __init__(
        self, ec2_client: Any, instances: Optional[List[Dict[str, Any]]] = None
    ) -> None:
        """
        Initializes the EC2InstanceWrapper with an EC2 client and optional
        instances.

```

```

        :param ec2_client: A Boto3 Amazon EC2 client. This client provides low-
level
        access to AWS EC2 services.
        :param instances: A list of dictionaries representing Boto3 Instance
objects. These are high-level objects that
        wrap instance actions.
        """
        self.ec2_client = ec2_client
        self.instances = instances or []

    @classmethod
    def from_client(cls) -> "EC2InstanceWrapper":
        """
        Creates an EC2InstanceWrapper instance with a default EC2 client.

        :return: An instance of EC2InstanceWrapper initialized with the default
EC2 client.
        """
        ec2_client = boto3.client("ec2")
        return cls(ec2_client)

    def get_instance_types(
        self, architecture: str = "x86_64", sizes: List[str] = ["*.micro",
        "/*.small"]
    ) -> List[Dict[str, Any]]:
        """
        Gets instance types that support the specified architecture and size.
        See https://docs.aws.amazon.com/AWSEC2/latest/APIReference/
API\_DescribeInstanceTypes.html
        for a list of allowable parameters.

        :param architecture: The architecture supported by instance types.
Default: 'x86_64'.
        :param sizes: The size of instance types. Default: '/*.micro', '/*.small',
        :return: A list of dictionaries representing instance types that support
the specified architecture and size.
        """
        try:
            inst_types = []
            paginator = self.ec2_client.get_paginator("describe_instance_types")
            for page in paginator.paginate(
                Filters=[
                    {

```

```
        "Name": "processor-info.supported-architecture",
        "Values": [architecture],
    },
    {"Name": "instance-type", "Values": sizes},
]
):
    inst_types += page["InstanceTypes"]
except ClientError as err:
    logger.error(
        f"Failed to get instance types: {architecture},
{'.'.join(map(str, sizes))}"
    )
    error_code = err.response["Error"]["Code"]
    if error_code == "InvalidParameterValue":
        logger.error(
            "Parameters are invalid. "
            "Ensure architecture and size strings conform to
DescribeInstanceTypes API reference."
        )
        raise
    else:
        return inst_types
```

- Para API obter detalhes, consulte a [DescribeInstanceTypes](#) Referência AWS SDK do Python (Boto3). API

Rust

SDK para Rust

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
/// List instance types that match an image's architecture and are free tier
eligible.
```

```
pub async fn list_instance_types(&self, image: &Image) ->
Result<Vec<InstanceType>, EC2Error> {
    let architecture = format!(
        "{}",
        image.architecture().ok_or_else(|| EC2Error::new(format!(
            "Image {:?} does not have a listed architecture",
            image.image_id()
        )))?
    );
    let free_tier_eligible_filter = Filter::builder()
        .name("free-tier-eligible")
        .values("false")
        .build();
    let supported_architecture_filter = Filter::builder()
        .name("processor-info.supported-architecture")
        .values(architecture)
        .build();
    let response = self
        .client
        .describe_instance_types()
        .filters(free_tier_eligible_filter)
        .filters(supported_architecture_filter)
        .send()
        .await?;

    Ok(response
        .instance_types
        .unwrap_or_default()
        .into_iter()
        .filter_map(|iti| iti.instance_type)
        .collect())
}
```

- Para API obter detalhes, consulte [DescribeInstanceTypes](#) a AWS SDK API referência do Rust.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use `DescribeInstances` com um AWS SDK ou CLI

Os exemplos de código a seguir mostram como usar o `DescribeInstances`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto nos seguintes exemplos de código:

- [Aprender os conceitos básicos](#)
- [Criar e gerenciar um serviço resiliente](#)

.NET

AWS SDK for .NET

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
/// <summary>
/// Get information about EC2 instances with a particular state.
/// </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>True if successful.</returns>
public async Task<bool> GetInstancesWithState(string state)
{
    try
    {
        // Filters the results of the instance list.
        var filters = new List<Filter>
        {
            new Filter
            {
                Name = $"instance-state-name",
                Values = new List<string> { state, },
            },
        };
        var request = new DescribeInstancesRequest { Filters = filters, };
```

```
Console.WriteLine($"\\nShowing instances with state {state}");
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}");
        }
    }

    return true;
}
catch (AmazonEC2Exception ec2Exception)
{
    if (ec2Exception.ErrorCode == "InvalidParameterValue")
    {
        _logger.LogError(
            $"Invalid parameter value for filtering instances.");
    }

    return false;
}
catch (Exception ex)
{
    Console.WriteLine($"Couldn't list instances because: {ex.Message}");
    return false;
}
}
```

- Para API obter detalhes, consulte [DescribeInstances](#) em AWS SDK for .NET API Referência.

Bash

AWS CLI com script Bash

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
#####
# 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 getopt 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 getopt "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

```



```
}

```

As funções utilitárias usadas neste exemplo.

```
#####
# 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
}

```

- Para API obter detalhes, consulte [DescribeInstances](#) na Referência de AWS CLI Comandos.

C++

SDK para C++

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```

//! Describe all Amazon Elastic Compute Cloud (Amazon EC2) instances associated
with an account.
/*!
\param clientConfiguration: AWS client configuration.
\return bool: Function succeeded.
*/
bool AwsDoc::EC2::describeInstances(
    const Aws::Client::ClientConfiguration &clientConfiguration) {
    Aws::EC2::EC2Client ec2Client(clientConfiguration);
    Aws::EC2::Model::DescribeInstancesRequest request;
    bool header = false;
    bool done = false;
    while (!done) {
        Aws::EC2::Model::DescribeInstancesOutcome 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;
}

return true;
}
```

- Para API obter detalhes, consulte [DescribeInstances](#) em AWS SDK for C++ API Referência.

CLI

AWS CLI

Exemplo 1: descrever uma instância

O exemplo `describe-instances` a seguir descreve a instância especificada.

```
aws ec2 describe-instances \
  --instance-ids i-1234567890abcdef0
```

Saída:

```
{
  "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-04a636d18e83cfac",
"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": "11111111111111111111111111111111"
```

```
    },
    "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"
}
]
}

```

Exemplo 2: para filtrar instâncias com o tipo especificado

O exemplo `describe-instances` a seguir usa filtros para definir o escopo dos resultados para instâncias do tipo especificado.

```

aws ec2 describe-instances \
  --filters Name=instance-type,Values=m5.large

```

Para obter um exemplo da saída, consulte o Exemplo 1.

Para obter mais informações, consulte [Listar e filtrar usando o CLI](#) no Guia EC2 do usuário da Amazon.

Exemplo 3: filtrar instâncias com o tipo e a zona de disponibilidade especificados

O exemplo `describe-instances` a seguir usa vários filtros para definir o escopo dos resultados para instâncias com o tipo especificado que também estão na zona de disponibilidade especificada.

```

aws ec2 describe-instances \
  --filters Name=instance-type,Values=t2.micro,t3.micro Name=availability-zone,Values=us-east-2c

```


Para obter um exemplo da saída, consulte o Exemplo 1.

Exemplo 4: Para filtrar instâncias com o tipo e a zona de disponibilidade especificados usando um JSON arquivo

O `describe-instances` exemplo a seguir usa um arquivo JSON de entrada para realizar a mesma filtragem do exemplo anterior. Quando os filtros ficam mais complicados, é mais fácil especificá-los em um JSON arquivo.

```
aws ec2 describe-instances \  
  --filters file://filters.json
```

Conteúdo de `filters.json`:

```
[  
  {  
    "Name": "instance-type",  
    "Values": ["t2.micro", "t3.micro"]  
  },  
  {  
    "Name": "availability-zone",  
    "Values": ["us-east-2c"]  
  }  
]
```

Para obter um exemplo da saída, consulte o Exemplo 1.

Exemplo 5: filtrar instâncias com a tag Proprietário especificada

O exemplo `describe-instances` a seguir usa filtros de tag para definir o escopo dos resultados para instâncias que têm uma tag com a chave de tag especificada (Proprietário), independentemente do valor da tag.

```
aws ec2 describe-instances \  
  --filters "Name=tag-key, Values=Owner"
```

Para obter um exemplo da saída, consulte o Exemplo 1.

Exemplo 6: filtrar instâncias com o valor especificado `my-team` da tag

O exemplo `describe-instances` a seguir usa filtros de tag para definir o escopo dos resultados para instâncias que têm uma tag com o valor especificado da tag (`my-team`), independentemente da chave da tag.

```
aws ec2 describe-instances \
  --filters "Name=tag-value,Values=my-team"
```

Para obter um exemplo da saída, consulte o Exemplo 1.

Exemplo 7: filtrar instâncias com a tag Proprietário e o valor `my-team` especificados

O exemplo `describe-instances` a seguir usa filtros de tag para definir o escopo dos resultados para instâncias que têm a tag especificada (`Owner=my-team`).

```
aws ec2 describe-instances \
  --filters "Name=tag:Owner,Values=my-team"
```

Para obter um exemplo da saída, consulte o Exemplo 1.

Exemplo 8: Para exibir somente a instância e a sub-rede IDs para todas as instâncias

Os `describe-instances` exemplos a seguir usam o `--query` parâmetro para exibir somente a instância e a sub-rede de todas IDs as instâncias, em JSON formato.

Linux e 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
```

Saída:

```
[
  {
```

```

    "Instance": "i-057750d42936e468a",
    "Subnet": "subnet-069beee9b12030077"
  },
  {
    "Instance": "i-001efd250faaa6ffa",
    "Subnet": "subnet-0b715c6b7db68927a"
  },
  {
    "Instance": "i-027552a73f021f3bd",
    "Subnet": "subnet-0250c25a1f4e15235"
  }
  ...
]

```

Exemplo 9: Para filtrar instâncias do tipo especificado e exibir somente sua instância IDs

O `describe-instances` exemplo a seguir usa filtros para definir o escopo dos resultados para instâncias do tipo especificado e o `--query` parâmetro para exibir somente a instância IDs.

```

aws ec2 describe-instances \
  --filters "Name=instance-type,Values=t2.micro" \
  --query "Reservations[*].Instances[*].[InstanceId]" \
  --output text

```

Saída:

```

i-031c0dc19de2fb70c
i-00d8bfff789a736b75
i-0b715c6b7db68927a
i-0626d4edd54f1286d
i-00b8ae04f9f99908e
i-0fc71c25d2374130c

```

Exemplo 10: filtrar instâncias do tipo especificado e exibir somente a instância IDs, a zona de disponibilidade e o valor da tag especificada

Os exemplos `describe-instances` a seguir exibem o ID da instância, a zona de disponibilidade e o valor da tag Name para instâncias que têm uma tag com o nome `tag-key`, em formato de tabela.

Linux e 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
```

Saída:

```
-----
|                               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   |
+-----+-----+-----+-----+
```

Exemplo 11: descrever instâncias em um grupo com posicionamento em partições

O exemplo `describe-instances` a seguir descreve a instância especificada. A saída inclui as informações de posicionamento da instância, o que contém o nome do grupo de posicionamento e o número da partição da instância.

```
aws ec2 describe-instances \
  --instance-ids i-0123a456700123456 \
  --query "Reservations[*].Instances[*].Placement"
```

Saída:

```
[
```

```
[
  {
    "AvailabilityZone": "us-east-1c",
    "GroupName": "HDFS-Group-A",
    "PartitionNumber": 3,
    "Tenancy": "default"
  }
]
```

Para obter mais informações, consulte [Descrição de instâncias em um grupo de posicionamento](#) no Guia EC2 do usuário da Amazon.

Exemplo 12: filtrar instâncias com o grupo de posicionamento e o número de partição especificados

O exemplo `describe-instances` a seguir filtra os resultados somente para as instâncias com o grupo de posicionamento e o número de partição especificados.

```
aws ec2 describe-instances \
  --filters "Name=placement-group-name,Values=HDFS-Group-A" "Name=placement-
  partition-number,Values=7"
```

A seguir, são mostradas somente as informações relevantes da saída.

```
"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"  
    }  
],
```

Para obter mais informações, consulte [Descrição de instâncias em um grupo de posicionamento](#) no Guia EC2 do usuário da Amazon.

Exemplo 13: filtrar instâncias configuradas para permitir o acesso às tags dos metadados da instância

O exemplo `describe-instances` a seguir filtra os resultados somente para as instâncias que estão configuradas para permitir o acesso às tags de instância nos metadados da instância.

```
aws ec2 describe-instances \  
  --filters "Name=metadata-options.instance-metadata-tags,Values=enabled" \  
  --query "Reservations[*].Instances[*].InstanceId" \  
  --output text
```

A saída esperada é mostrada a seguir.

```
i-1234567890abcdefg  
i-abcdefg1234567890  
i-111111111aaaaaaaa  
i-aaaaaaaa111111111
```

Para obter mais informações, consulte [Trabalhar com tags de instância em metadados de instância](#) no Guia do EC2 usuário da Amazon.

- Para API obter detalhes, consulte [DescribeInstances](#) na Referência de AWS CLI Comandos.

Java

SDK para Java 2.x

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```

/**
 * Asynchronously describes an AWS EC2 image with the specified image ID.
 *
 * @param imageId the ID of the image to be described
 * @return a {@link CompletableFuture} that, when completed, contains the ID
of the described image
 * @throws RuntimeException if no images are found with the provided image
ID, or if an error occurs during the AWS API call
 */
public CompletableFuture<String> describeImageAsync(String imageId) {
    DescribeImagesRequest imagesRequest = DescribeImagesRequest.builder()
        .imageIds(imageId)
        .build();

    AtomicReference<String> imageIdRef = new AtomicReference<>();
    DescribeImagesPublisher paginator =
getAsyncClient().describeImagesPaginator(imagesRequest);
    return paginator.subscribe(response -> {
        response.images().stream()
            .filter(image -> image.imageId().equals(imageId))
            .findFirst()
            .ifPresent(image -> {
                logger.info("The description of the image is " +
image.description());
                logger.info("The name of the image is " + image.name());
                imageIdRef.set(image.imageId());
            });
    }).thenApply(v -> {
        String id = imageIdRef.get();
        if (id == null) {
            throw new RuntimeException("No images found with the provided
image ID.");
        }
        return id;
    }).exceptionally(ex -> {
        logger.info("Failed to describe image: " + ex.getMessage());
        throw new RuntimeException("Failed to describe image", ex);
    });
}

```

- Para API obter detalhes, consulte [DescribeInstances](#) em AWS SDK for Java 2.x API Referência.

JavaScript

SDK para JavaScript (v3)

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
import { EC2Client, paginateDescribeInstances } from "@aws-sdk/client-ec2";

/**
 * List all of your EC2 instances running with the provided architecture that
 * were launched in the past month.
 * @param {{ pageSize: string, architectures: string[] }} options
 */
export const main = async ({ pageSize, architectures }) => {
  pageSize = Number.parseInt(pageSize);
  const client = new EC2Client({});
  const d = new Date();
  const year = d.getFullYear();
  const month = `0${d.getMonth() + 1}`.slice(-2);
  const launchTimePattern = `${year}-${month}-*`;

  const paginator = paginateDescribeInstances(
    {
      client,
      pageSize,
    },
    {
      Filters: [
        { Name: "architecture", Values: architectures },
        { Name: "instance-state-name", Values: ["running"] },
        {
          Name: "launch-time",
          Values: [launchTimePattern],
        },
      ],
    },
  );
};
```



```
try {
  /**
   * @type {import('@aws-sdk/client-ec2').Instance[]}
   */
  const instanceList = [];
  for await (const page of paginator) {
    const { Reservations } = page;
    for (const reservation of Reservations) {
      instanceList.push(...reservation.Instances);
    }
  }
  console.log(
    `Running instances launched this month:\n\n${instanceList.map((instance) =>
instance.InstanceId).join("\n")}` ,
  );
} catch (caught) {
  if (caught instanceof Error && caught.name === "InvalidParameterValue") {
    console.warn(`${caught.message}.`);
  } else {
    throw caught;
  }
}
};
```

- Para API obter detalhes, consulte [DescribeInstances](#) em AWS SDK for JavaScript API Referência.

Kotlin

SDK para Kotlin

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
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}")
            }
        }
    }
}

```

- Para API obter detalhes, consulte a [DescribeInstances](#) referência AWS SDK do Kotlin API.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Esse exemplo descreve a instância especificada.

```
(Get-EC2Instance -InstanceId i-12345678).Instances
```

Saída:

```

AmiLaunchIndex      : 0
Architecture        : x86_64
BlockDeviceMappings : {/dev/sda1}
ClientToken          : T1eEy1448154045270
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
```

Exemplo 2: Este exemplo descreve todas as suas instâncias na região atual, agrupadas por reserva. Para ver os detalhes da instância, expanda a coleção de instâncias em cada objeto de reserva.

```
Get-EC2Instance
```

Saída:

```
GroupNames    : {}  
Groups        : {}  
Instances     : {}  
OwnerId       : 123456789012  
RequesterId   : 226008221399  
ReservationId : r-c5df370c  
  
GroupNames    : {}
```

```

Groups      : {}
Instances   : {}
OwnerId     : 123456789012
RequesterId : 854251627541
ReservationId : r-63e65bab
...

```

Exemplo 3: Este exemplo ilustra o uso de um filtro para consultar EC2 instâncias em uma sub-rede específica de a. VPC

```
(Get-EC2Instance -Filter @{{Name="vpc-id";Values="vpc-1a2bc34d"}},{Name="subnet-id";Values="subnet-1a2b3c4d"}).Instances
```

Saída:

```

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

```

- Para API obter detalhes, consulte [DescribeInstances](#) em Referência de AWS Tools for PowerShell cmdlet.

Python

SDK para Python (Boto3)

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```

class EC2InstanceWrapper:
    """Encapsulates Amazon Elastic Compute Cloud (Amazon EC2) instance actions
    using the client interface."""

```

```
def __init__(
    self, ec2_client: Any, instances: Optional[List[Dict[str, Any]]] = None
) -> None:
    """
    Initializes the EC2InstanceWrapper with an EC2 client and optional
    instances.

    :param ec2_client: A Boto3 Amazon EC2 client. This client provides low-
level
                        access to AWS EC2 services.
    :param instances: A list of dictionaries representing Boto3 Instance
objects. These are high-level objects that
                        wrap instance actions.
    """
    self.ec2_client = ec2_client
    self.instances = instances or []

    @classmethod
    def from_client(cls) -> "EC2InstanceWrapper":
        """
        Creates an EC2InstanceWrapper instance with a default EC2 client.

        :return: An instance of EC2InstanceWrapper initialized with the default
EC2 client.
        """
        ec2_client = boto3.client("ec2")
        return cls(ec2_client)

    def display(self, state_filter: Optional[str] = "running") -> None:
        """
        Displays information about instances, filtering by the specified state.

        :param state_filter: The instance state to include in the output. Only
instances in this state
                            will be displayed. Default is 'running'. Example
states: 'running', 'stopped'.
        """
        if not self.instances:
            logger.info("No instances to display.")
            return

        instance_ids = [instance["InstanceId"] for instance in self.instances]
```

```

paginator = self.ec2_client.get_paginator("describe_instances")
page_iterator = paginator.paginate(InstanceIds=instance_ids)

try:
    for page in page_iterator:
        for reservation in page["Reservations"]:
            for instance in reservation["Instances"]:
                instance_state = instance["State"]["Name"]

                # Apply the state filter (default is 'running')
                if state_filter and instance_state != state_filter:
                    continue # Skip this instance if it doesn't match
the filter

                # Create a formatted string with instance details
                instance_info = (
                    f"• ID: {instance['InstanceId']}\n"
                    f"• Image ID: {instance['ImageId']}\n"
                    f"• Instance type: {instance['InstanceType']}\n"
                    f"• Key name: {instance['KeyName']}\n"
                    f"• VPC ID: {instance['VpcId']}\n"
                    f"• Public IP: {instance.get('PublicIpAddress', 'N/A')}\n"
                    f"• State: {instance_state}"
                )
                print(instance_info)

except ClientError as err:
    logger.error(
        f"Failed to display instance(s). : {' '.join(map(str,
instance_ids))}"
    )
    error_code = err.response["Error"]["Code"]
    if error_code == "InvalidInstanceID.NotFound":
        logger.error(
            "One or more instance IDs do not exist. "
            "Please verify the instance IDs and try again."
        )
        raise

```

- Para API obter detalhes, consulte a [DescribeInstances](#) Referência AWS SDK do Python (Boto3). API

Ruby

SDK para Ruby

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
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
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__
```

- Para API obter detalhes, consulte [DescribeInstances](#) em AWS SDK for Ruby API Referência.

Rust

SDK para Rust

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

Recupere detalhes de uma EC2 instância.

```
pub async fn describe_instance(&self, instance_id: &str) -> Result<Instance, EC2Error> {
    let response = self
        .client
        .describe_instances()
        .instance_ids(instance_id)
        .send()
        .await?;

    let instance = response
        .reservations()
        .first()
```



```

        .ok_or_else(|| EC2Error::new(format!("No instance reservations for
{instance_id}")))?
        .instances()
        .first()
        .ok_or_else(|| {
            EC2Error::new(format!("No instances in reservation for
{instance_id}"))
        })?;

    Ok(instance.clone())
}

```

Depois de criar uma EC2 instância, recupere e armazene seus detalhes.

```

/// Create an EC2 instance with the given ID on a given type, using a
/// generated KeyPair and applying a list of security groups.
pub async fn create(
    &mut self,
    ec2: &EC2,
    image_id: &str,
    instance_type: InstanceType,
    key_pair: &KeyPairInfo,
    security_groups: Vec<&SecurityGroup>,
) -> Result<(), EC2Error> {
    let instance_id = ec2
        .create_instance(image_id, instance_type, key_pair, security_groups)
        .await?;
    let instance = ec2.describe_instance(&instance_id).await?;
    self.instance = Some(instance);
    Ok(())
}

```

- Para API obter detalhes, consulte [DescribeInstances](#) a AWS SDKAPIreferência do Rust.

SAP ABAP

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```

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( ).
        ENDLLOOP.
    ENDLLOOP.
    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.

```

- Para API obter detalhes, consulte [DescribeInstances AWSSDK para SAP ABAP API](#) referência.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use `DescribeInternetGateways` com um CLI

Os exemplos de código a seguir mostram como usar o `DescribeInternetGateways`.

CLI

AWS CLI

Para descrever um gateway de internet

O `describe-internet-gateways` exemplo a seguir descreve o gateway de internet especificado.

```
aws ec2 describe-internet-gateways \
  --internet-gateway-ids igw-0d0fb496b3EXAMPLE
```

Saída:

```
{
  "InternetGateways": [
    {
      "Attachments": [
        {
          "State": "available",
          "VpcId": "vpc-0a60eb65b4EXAMPLE"
        }
      ],
      "InternetGatewayId": "igw-0d0fb496b3EXAMPLE",
      "OwnerId": "123456789012",
      "Tags": [
        {
          "Key": "Name",
          "Value": "my-igw"
        }
      ]
    }
  ]
}
```

Para obter mais informações, consulte [Gateways de Internet](#) no Guia do VPC usuário da Amazon.

- Para API obter detalhes, consulte [DescribeInternetGateways](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo descreve o gateway de Internet especificado.

```
Get-EC2InternetGateway -InternetGatewayId igw-1a2b3c4d
```

Saída:

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

Exemplo 2: Este exemplo descreve todos os seus gateways de Internet.

```
Get-EC2InternetGateway
```

Saída:

Attachments	InternetGatewayId	Tags
-----	-----	----
{vpc-1a2b3c4d}	igw-1a2b3c4d	{}
{}	igw-2a3b4c5d	{}

- Para API obter detalhes, consulte [DescribeInternetGateways](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **DescribeKeyPairs** com um AWS SDK ou CLI

Os exemplos de código a seguir mostram como usar o `DescribeKeyPairs`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação no contexto no seguinte exemplo de código:

- [Aprender os conceitos básicos](#)

.NET

AWS SDK for .NET

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
/// <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)
{
    try
    {
        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();
    }
    catch (AmazonEC2Exception ec2Exception)
    {
        if (ec2Exception.ErrorCode == "InvalidKeyPair.NotFound")
        {
            _logger.LogError(
```

```

        $"A key pair called {keyPairName} does not exist.");
    }

    throw;
}
catch (Exception ex)
{
    _logger.LogError(
        $"An error occurred while describing the key pair.:
{ex.Message}");
    throw;
}
}
}

```

- Para API obter detalhes, consulte [DescribeKeyPairs](#) em AWS SDK for .NET API Referência.

Bash

AWS CLI com script Bash

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```

#####
# 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 getopt 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 getopt "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
}
```

As funções utilitárias usadas neste exemplo.

```
#####
# 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
}
}
```


- Para API obter detalhes, consulte [DescribeKeyPairs](#) na Referência de AWS CLI Comandos.

C++

SDK para C++

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```

//! Describe all Amazon Elastic Compute Cloud (Amazon EC2) instance key pairs.
/*!
 \param clientConfiguration: AWS client configuration.
 \return bool: Function succeeded.
 */
bool AwsDoc::EC2::describeKeyPairs(
    const Aws::Client::ClientConfiguration &clientConfiguration) {
    Aws::EC2::EC2Client ec2Client(clientConfiguration);
    Aws::EC2::Model::DescribeKeyPairsRequest request;

    Aws::EC2::Model::DescribeKeyPairsOutcome 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;
    }
}

```

```
    return outcome.IsSuccess();  
}
```

- Para API obter detalhes, consulte [DescribeKeyPairs](#) em AWS SDK for C++ API Referência.

CLI

AWS CLI

Exibir um par de chaves

O exemplo `describe-key-pairs` a seguir mostra as informações do par de chaves especificado.

```
aws ec2 describe-key-pairs \  
  --key-names my-key-pair
```

Saída:

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

Para obter mais informações, consulte [Descrever as chaves públicas](#) no Guia EC2 do usuário da Amazon.

- Para API obter detalhes, consulte [DescribeKeyPairs](#) na Referência de AWS CLI Comandos.

Java

SDK para Java 2.x

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
/**
 * Asynchronously describes the key pairs associated with the current AWS
 account.
 *
 * @return a {@link CompletableFuture} containing the {@link
 DescribeKeyPairsResponse} object, which provides
 * information about the key pairs.
 */
public CompletableFuture<DescribeKeyPairsResponse> describeKeysAsync() {
    CompletableFuture<DescribeKeyPairsResponse> responseFuture =
getAsyncClient().describeKeyPairs();
    responseFuture.whenComplete((response, exception) -> {
        if (exception != null) {
            throw new RuntimeException("Failed to describe key pairs: " +
exception.getMessage(), exception);
        }
    });

    return responseFuture;
}
```

- Para API obter detalhes, consulte [DescribeKeyPair](#) em AWS SDK for Java 2.x API Referência.

JavaScript

SDK para JavaScript (v3)

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
import { DescribeKeyPairsCommand, EC2Client } from "@aws-sdk/client-ec2";

/**
 * List all key pairs in the current AWS account.
 * @param {{ dryRun: boolean }}
 */
export const main = async ({ dryRun }) => {
  const client = new EC2Client({});
  const command = new DescribeKeyPairsCommand({ DryRun: dryRun });

  try {
    const { KeyPairs } = await client.send(command);
    const keyPairList = KeyPairs.map(
      (kp) => ` • ${kp.KeyPairId}: ${kp.KeyName}`,
    ).join("\n");
    console.log("The following key pairs were found in your account:");
    console.log(keyPairList);
  } catch (caught) {
    if (caught instanceof Error && caught.name === "DryRunOperation") {
      console.log(`${caught.message}`);
    } else {
      throw caught;
    }
  }
};
```

- Para API obter detalhes, consulte [DescribeKeyPair](#) em AWS SDK for JavaScript API Referência.

Kotlin

SDK para Kotlin

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
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}")
        }
    }
}
```

- Para API obter detalhes, consulte a [DescribeKeyPairs](#) referência AWS SDK do Kotlin API.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo descreve o par de chaves especificado.

```
Get-EC2KeyPair -KeyName my-key-pair
```

Saída:

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

Exemplo 2: Este exemplo descreve todos os seus pares de chaves.

```
Get-EC2KeyPair
```

- Para API obter detalhes, consulte [DescribeKeyPair](#) em Referência de AWS Tools for PowerShell cmdlet.

Python

SDK para Python (Boto3)

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
class KeyPairWrapper:
    """
    Encapsulates Amazon Elastic Compute Cloud (Amazon EC2) key pair actions.
    This class provides methods to create, list, and delete EC2 key pairs.
    """

    def __init__(
        self,
        ec2_client: boto3.client,
        key_file_dir: Union[tempfile.TemporaryDirectory, str],
        key_pair: Optional[dict] = None,
    ):
        """
        Initializes the KeyPairWrapper with the specified EC2 client, key file
        directory,
        and an optional key pair.

        :param ec2_client: A Boto3 Amazon EC2 client. This client provides low-
        level
            access to AWS EC2 services.
        :param key_file_dir: The folder where the private key information is
        stored.
            This should be a secure folder.
        :param key_pair: A dictionary representing the Boto3 KeyPair object.
            This is a high-level object that wraps key pair actions.
        Optional.
        """
        self.ec2_client = ec2_client
```

```
self.key_pair = key_pair
self.key_file_path: Optional[str] = None
self.key_file_dir = key_file_dir

@classmethod
def from_client(cls) -> "KeyPairWrapper":
    """
    Class method to create an instance of KeyPairWrapper using a new EC2
client
and a temporary directory for storing key files.

:return: An instance of KeyPairWrapper.
    """
    ec2_client = boto3.client("ec2")
    return cls(ec2_client, tempfile.TemporaryDirectory())

def list(self, limit: Optional[int] = None) -> None:
    """
    Displays a list of key pairs for the current account.

    WARNING: Results are not paginated.

    :param limit: The maximum number of key pairs to list. If not specified,
all key pairs will be listed.
    :raises ClientError: If there is an error in listing the key pairs.
    """
    try:
        response = self.ec2_client.describe_key_pairs()
        key_pairs = response.get("KeyPairs", [])

        if limit:
            key_pairs = key_pairs[:limit]

        for key_pair in key_pairs:
            logger.info(
                f"Found {key_pair['KeyType']} key '{key_pair['KeyName']}'
with fingerprint:"
            )
            logger.info(f"\t{key_pair['KeyFingerprint']}")
    except ClientError as err:
        logger.error(f"Failed to list key pairs: {str(err)}")
        raise
```

- Para API obter detalhes, consulte a [DescribeKeyPairs](#) Referência AWS SDK do Python (Boto3). API

Rust

SDK para Rust

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
pub async fn list_key_pair(&self) -> Result<Vec<KeyPairInfo>, EC2Error> {
    let output = self.client.describe_key_pairs().send().await?;
    Ok(output.key_pairs.unwrap_or_default())
}
```

- Para API obter detalhes, consulte [DescribeKeyPairs](#) a AWS SDK API referência do Rust.

SAP ABAP

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
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.
```

- Para API obter detalhes, consulte [DescribeKeyPairs AWS SDK](#) para SAP ABAP API referência.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **DescribeNetworkAcls** com um CLI

Os exemplos de código a seguir mostram como usar o DescribeNetworkAcls.

CLI

AWS CLI

Para descrever sua rede ACLs

O describe-network-acls exemplo a seguir recupera detalhes sobre sua rede ACLs.

```
aws ec2 describe-network-acls
```

Saída:

```
{
  "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"
}
]
}

```

Para obter mais informações, consulte [Rede ACLs](#) no Guia AWS VPC do Usuário.

- Para API obter detalhes, consulte [DescribeNetworkAcls](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo descreve a rede especificadaACL.

```
Get-EC2NetworkAcl -NetworkAclId acl-12345678
```

Saída:

```

Associations : {aclassoc-1a2b3c4d}
Entries      : {Amazon.EC2.Model.NetworkAclEntry,
               Amazon.EC2.Model.NetworkAclEntry}
IsDefault    : False
NetworkAclId : acl-12345678
Tags         : {Name}
VpcId        : vpc-12345678

```

Exemplo 2: Este exemplo descreve as regras da rede especificadaACL.

```
(Get-EC2NetworkAcl -NetworkAclId acl-12345678).Entries
```

Saída:

```
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
```

Exemplo 3: Este exemplo descreve toda a sua redeACLs.

```
Get-EC2NetworkAcl
```

- Para API obter detalhes, consulte [DescribeNetworkAcls](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **DescribeNetworkInterfaceAttribute** com um CLI

Os exemplos de código a seguir mostram como usar o `DescribeNetworkInterfaceAttribute`.

CLI

AWS CLI

Para descrever o atributo de anexo de uma interface de rede

Este exemplo de comando descreve o `attachment` atributo da interface de rede especificada.

Comando:

```
aws ec2 describe-network-interface-attribute --network-interface-id eni-686ea200
--attribute attachment
```

Saída:

```
{
  "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"
  }
}
```

Para descrever o atributo de descrição de uma interface de rede

Este exemplo de comando descreve o `description` atributo da interface de rede especificada.

Comando:

```
aws ec2 describe-network-interface-attribute --network-interface-id eni-686ea200
--attribute description
```

Saída:

```
{
  "NetworkInterfaceId": "eni-686ea200",
  "Description": {
    "Value": "My description"
  }
}
```

Para descrever o `groupSet` atributo de uma interface de rede

Este exemplo de comando descreve o `groupSet` atributo da interface de rede especificada.

Comando:

```
aws ec2 describe-network-interface-attribute --network-interface-id eni-686ea200
--attribute groupSet
```

Saída:

```
{
  "NetworkInterfaceId": "eni-686ea200",
  "Groups": [
    {
      "GroupName": "my-security-group",
      "GroupId": "sg-903004f8"
    }
  ]
}
```

Para descrever o `sourceDestCheck` atributo de uma interface de rede

Este exemplo de comando descreve o `sourceDestCheck` atributo da interface de rede especificada.

Comando:

```
aws ec2 describe-network-interface-attribute --network-interface-id eni-686ea200
--attribute sourceDestCheck
```

Saída:

```
{
  "NetworkInterfaceId": "eni-686ea200",
  "SourceDestCheck": {
    "Value": true
  }
}
```

- Para API obter detalhes, consulte [DescribeNetworkInterfaceAttribute](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo descreve a interface de rede especificada.

```
Get-EC2NetworkInterfaceAttribute -NetworkInterfaceId eni-12345678 -Attribute Attachment
```

Saída:

```
Attachment           : Amazon.EC2.Model.NetworkInterfaceAttachment
```

Exemplo 2: Este exemplo descreve a interface de rede especificada.

```
Get-EC2NetworkInterfaceAttribute -NetworkInterfaceId eni-12345678 -Attribute Description
```

Saída:

```
Description          : My description
```

Exemplo 3: Este exemplo descreve a interface de rede especificada.

```
Get-EC2NetworkInterfaceAttribute -NetworkInterfaceId eni-12345678 -Attribute GroupSet
```

Saída:

```
Groups                : {my-security-group}
```

Exemplo 4: Este exemplo descreve a interface de rede especificada.

```
Get-EC2NetworkInterfaceAttribute -NetworkInterfaceId eni-12345678 -Attribute SourceDestCheck
```

Saída:

```
SourceDestCheck      : True
```


- Para API obter detalhes, consulte [DescribeNetworkInterfaceAttribute](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use `DescribeNetworkInterfaces` com um CLI

Os exemplos de código a seguir mostram como usar o `DescribeNetworkInterfaces`.

CLI

AWS CLI

Para descrever suas interfaces de rede

Este exemplo descreve todas as suas interfaces de rede.

Comando:

```
aws ec2 describe-network-interfaces
```

Saída:

```
{
  "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"
    }
]
}

```

Este exemplo descreve interfaces de rede que têm uma tag com a chave Purpose e o valor Prod.

Comando:

```
aws ec2 describe-network-interfaces --filters Name=tag:Purpose,Values=Prod
```

Saída:

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

- Para API obter detalhes, consulte [DescribeNetworkInterfaces](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo descreve a interface de rede especificada.

```
Get-EC2NetworkInterface -NetworkInterfaceId eni-12345678
```

Saída:

```
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  
PrivateAddresses : {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
```

Exemplo 2: Este exemplo descreve todas as suas interfaces de rede.

```
Get-EC2NetworkInterface
```

- Para API obter detalhes, consulte [DescribeNetworkInterfaces](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **DescribePlacementGroups** com um CLI

Os exemplos de código a seguir mostram como usar o `DescribePlacementGroups`.

CLI

AWS CLI

Para descrever seus grupos de colocação

Este exemplo de comando descreve todos os seus grupos de posicionamento.

Comando:

```
aws ec2 describe-placement-groups
```

Saída:

```
{
  "PlacementGroups": [
    {
      "GroupName": "my-cluster",
      "State": "available",
      "Strategy": "cluster"
    },
    ...
  ]
}
```

- Para API obter detalhes, consulte [DescribePlacementGroups](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo descreve o grupo de posicionamento especificado.

```
Get-EC2PlacementGroup -GroupName my-placement-group
```

Saída:

GroupName	State	Strategy
-----	-----	-----
my-placement-group	available	cluster

- Para API obter detalhes, consulte [DescribePlacementGroups](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **DescribePrefixLists** com um CLI

Os exemplos de código a seguir mostram como usar o `DescribePrefixLists`.

CLI

AWS CLI

Para descrever listas de prefixos

Este exemplo lista todas as listas de prefixos disponíveis para a região.

Comando:

```
aws ec2 describe-prefix-lists
```

Saída:

```
{
  "PrefixLists": [
    {
```

```

    "PrefixListName": "com.amazonaws.us-east-1.s3",
    "Cidrs": [
      "54.231.0.0/17"
    ],
    "PrefixListId": "pl-63a5400a"
  }
]
}

```

- Para API obter detalhes, consulte [DescribePrefixLists](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo busca o disponível Serviços da AWS em um formato de lista de prefixos para a região

```
Get-EC2PrefixList
```

Saída:

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

- Para API obter detalhes, consulte [DescribePrefixLists](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **DescribeRegions** com um AWS SDK ou CLI

Os exemplos de código a seguir mostram como usar o DescribeRegions.

C++

SDK para C++

 Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
#!/ Describe all Amazon Elastic Compute Cloud (Amazon EC2) Regions.
/*!
  \param clientConfiguration: AWS client configuration.
  \return bool: Function succeeded.
 */
bool AwsDoc::EC2::describeRegions(
    const Aws::Client::ClientConfiguration &clientConfiguration) {
    Aws::EC2::EC2Client ec2Client(clientConfiguration);

    Aws::EC2::Model::DescribeRegionsRequest request;
    Aws::EC2::Model::DescribeRegionsOutcome outcome =
    ec2Client.DescribeRegions(request);
    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;
    }

    std::cout << std::endl;

    return outcome.IsSuccess();
}
```

```
}
```

- Para API obter detalhes, consulte [DescribeRegions](#) em AWS SDK for C++ API Referência.

CLI

AWS CLI

Exemplo 1: descrever todas as suas regiões habilitadas

O exemplo `describe-regions` a seguir descreve todas as regiões habilitadas para a sua conta.

```
aws ec2 describe-regions
```

Saída:

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

Para obter mais informações, consulte [Regiões e zonas](#) no Guia EC2 do usuário da Amazon.

Exemplo 2: para descrever as regiões habilitadas com um endpoint cujo nome contém uma string específica

O exemplo `describe-regions` a seguir descreve todas as regiões que você habilitou e que têm a string “us” no endpoint.

```
aws ec2 describe-regions \
  --filters "Name=endpoint,Values=*us*"
```

Saída:

```
{
  "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"
    }
]
}

```

Para obter mais informações, consulte [Regiões e zonas](#) no Guia EC2 do usuário da Amazon.

Exemplo 3: descrever todas as regiões

O exemplo `describe-regions` a seguir descreve todas as regiões disponíveis, incluindo as que estão desabilitadas.

```

aws ec2 describe-regions \
  --all-regions

```

Saída:

```

{
  "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"
}
]
}
```

Para obter mais informações, consulte [Regiões e zonas](#) no Guia EC2 do usuário da Amazon.

Exemplo 4: listar somente os nomes das regiões

O exemplo `describe-regions` a seguir usa o parâmetro `--query` para filtrar a saída e retornar somente os nomes das regiões como texto.

```
aws ec2 describe-regions \  
  --all-regions \  
  --query "Regions[].{Name:RegionName}" \  
  --output text
```

Saída:

```
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
```

Para obter mais informações, consulte [Regiões e zonas](#) no Guia EC2 do usuário da Amazon.

- Para API obter detalhes, consulte [DescribeRegions](#) na Referência de AWS CLI Comandos.

JavaScript

SDK para JavaScript (v3)

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).


```
import { DescribeRegionsCommand, EC2Client } from "@aws-sdk/client-ec2";

/**
 * List all available AWS regions.
 * @param {{ regionNames: string[], includeOptInRegions: boolean }} options
 */
export const main = async ({ regionNames, includeOptInRegions }) => {
  const client = new EC2Client({});
  const command = new DescribeRegionsCommand({
    // By default this command will not show regions that require you to opt-in.
    // When AllRegions is true, even the regions that require opt-in will be
    returned.
    AllRegions: includeOptInRegions,
    // You can omit the Filters property if you want to get all regions.
    Filters: regionNames?.length
      ? [
        {
          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: regionNames,
        },
      ],
      : undefined,
  });

  try {
    const { Regions } = await client.send(command);
    const regionsList = Regions.map((reg) => ` • ${reg.RegionName}`);
    console.log("Found regions:");
    console.log(regionsList.join("\n"));
  } catch (caught) {
    if (caught instanceof Error && caught.name === "DryRunOperation") {
      console.log(`${caught.message}`);
    } else {
      throw caught;
    }
  }
};
```

- Para API obter detalhes, consulte [DescribeRegions](#) em AWS SDK for JavaScript API Referência.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo descreve as regiões que estão disponíveis para você.

```
Get-EC2Region
```

Saída:

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

- Para API obter detalhes, consulte [DescribeRegions](#) em Referência de AWS Tools for PowerShell cmdlet.

Ruby

SDK para Ruby

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
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__
```

- Para API obter detalhes, consulte [DescribeRegions](#) em AWS SDK for Ruby API Referência.

Rust

SDK para Rust

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
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(())
}
```

- Para API obter detalhes, consulte [DescribeRegions](#) a AWS SDK API Referência do Rust.

SAP ABAP

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
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.
```

- Para API obter detalhes, consulte [DescribeRegions AWS SDK para SAP ABAP API referência](#).

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **DescribeRouteTables** com um AWS SDK ou CLI

Os exemplos de código a seguir mostram como usar o `DescribeRouteTables`.

CLI

AWS CLI

Para descrever suas tabelas de rotas

O `describe-route-tables` exemplo a seguir recupera os detalhes sobre suas tabelas de rotas.

```
aws ec2 describe-route-tables
```

Saída:

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

Para obter mais informações, consulte [Trabalhando com tabelas de rotas](#) no Guia AWS VPC do usuário.

- Para API obter detalhes, consulte [DescribeRouteTables](#) na Referência de AWS CLI Comandos.

PHP

SDK para PHP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
/**
 * @param array $routeTableIds
 * @param array $filters
 * @return array
 */
public function describeRouteTables(array $routeTableIds = [], array $filters
= []): array
{
    $parameters = [];
    if($routeTableIds){
        $parameters['RouteTableIds'] = $routeTableIds;
    }
    if($filters){
        $parameters['Filters'] = $filters;
    }
}
```

```
    }
    try {
        $paginator = $this->ec2Client->getPaginator("DescribeRouteTables",
$parameters);
        $contents = [];
        foreach ($paginator as $result) {
            foreach ($result['RouteTables'] as $object) {
                $contents[] = $object['RouteTableId'];
            }
        }
    } catch (Ec2Exception $caught){
        echo "There was a problem paginating the results of
DescribeRouteTables: {$caught->getAwsErrorMessage()}\n";
        throw $caught;
    }
    return $contents;
}
```

- Para API obter detalhes, consulte [DescribeRouteTables](#) em AWS SDK for PHP API Referência.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo descreve todas as suas tabelas de rotas.

```
Get-EC2RouteTable
```

Saída:

```
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 :
```

Exemplo 2: Este exemplo retorna detalhes da tabela de rotas especificada.

```
Get-EC2RouteTable -RouteTableId rtb-1a2b3c4d
```

Exemplo 3: Este exemplo descreve as tabelas de rotas para o especificadoVPC.

```
Get-EC2RouteTable -Filter @{ Name="vpc-id"; Values="vpc-1a2b3c4d" }
```

Saída:

```
Associations    : {rtbassoc-12345678}
PropagatingVgws : {}
Routes          : {, }
RouteTableId    : rtb-1a2b3c4d
Tags            : {}
VpcId           : vpc-1a2b3c4d
```

- Para API obter detalhes, consulte [DescribeRouteTables](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **DescribeScheduledInstanceAvailability** com um CLI

Os exemplos de código a seguir mostram como usar o `DescribeScheduledInstanceAvailability`.

CLI

AWS CLI

Para descrever um cronograma disponível

Este exemplo descreve uma programação que ocorre toda semana no domingo, começando na data especificada.

Comando:

```
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
```

Saída:

```
{  
  "ScheduledInstanceAvailabilitySet": [  
    {  
      "AvailabilityZone": "us-west-2b",  
      "TotalScheduledInstanceHours": 1219,  
      "PurchaseToken": "eyJ2IjoiMSIsInMiOiJEsImMiOi...",  
      "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"  
    },  
    ...  
  ]  
}
```

```
}
```

Para restringir os resultados, você pode adicionar filtros que especificam o sistema operacional, a rede e o tipo de instância.

Comando:

```
--filters nome=plataforma, valores=Linux/ nome=plataforma de rede, valores= - nome=tipo de
instância, UNIX valores=C4.large EC2 VPC
```

- Para API obter detalhes, consulte [DescribeScheduledInstanceAvailability](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo descreve uma programação que ocorre toda semana no domingo, começando na data especificada.

```
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
```

Saída:

```
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
```

...

Exemplo 2: Para restringir os resultados, você pode adicionar filtros para critérios como sistema operacional, rede e tipo de instância.

```
-Filter @{ Name="platform";Values="Linux/UNIX" },@{ Name="network-  
platform";Values="EC2-VPC" },@{ Name="instance-type";Values="c4.large" }
```

- Para API obter detalhes, consulte [DescribeScheduledInstanceAvailability](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use `DescribeScheduledInstances` com um CLI

Os exemplos de código a seguir mostram como usar o `DescribeScheduledInstances`.

CLI

AWS CLI

Para descrever suas instâncias programadas

Este exemplo descreve a instância agendada especificada.

Comando:

```
aws ec2 describe-scheduled-instances --scheduled-instance-  
ids sci-1234-1234-1234-123456789012
```

Saída:

```
{  
  "ScheduledInstanceSet": [  
    {  
      "AvailabilityZone": "us-west-2b",  
      "ScheduledInstanceId": "sci-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"
  }
]
}

```

Este exemplo descreve todas as suas instâncias programadas.

Comando:

```
aws ec2 describe-scheduled-instances
```

- Para API obter detalhes, consulte [DescribeScheduledInstances](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo descreve a instância agendada especificada.

```
Get-EC2ScheduledInstance -ScheduledInstanceId
sci-1234-1234-1234-1234-123456789012
```

Saída:

```
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
```

Exemplo 2: Este exemplo descreve todas as suas instâncias programadas.

```
Get-EC2ScheduledInstance
```

- Para API obter detalhes, consulte [DescribeScheduledInstances](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **DescribeSecurityGroups** com um AWS SDK ou CLI

Os exemplos de código a seguir mostram como usar o DescribeSecurityGroups.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação no contexto no seguinte exemplo de código:

- [Aprender os conceitos básicos](#)

.NET

AWS SDK for .NET

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
/// <summary>
/// Retrieve information for one or all Amazon EC2 security group.
/// </summary>
/// <param name="groupId">The optional Id of a specific Amazon EC2 security
group.</param>
/// <returns>A list of security group information.</returns>
public async Task<List<SecurityGroup>> DescribeSecurityGroups(string groupId)
{
    try
    {
        var securityGroups = new List<SecurityGroup>();
        var request = new DescribeSecurityGroupsRequest();

        if (!string.IsNullOrEmpty(groupId))
        {
            var groupIds = new List<string> { groupId };
            request.GroupIds = groupIds;
        }

        var paginatorForSecurityGroups =
            _amazonEC2.Paginators.DescribeSecurityGroups(request);

        await foreach (var securityGroup in
paginatorForSecurityGroups.SecurityGroups)
        {
            securityGroups.Add(securityGroup);
        }

        return securityGroups;
    }
    catch (AmazonEC2Exception ec2Exception)
```

```
    {
        if (ec2Exception.ErrorCode == "InvalidGroup.NotFound")
        {
            _logger.LogError(
                $"A security group {groupId} does not exist.");
        }

        throw;
    }
    catch (Exception ex)
    {
        _logger.LogError(
            {ex.Message});
        throw;
    }
}

/// <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.WriteLine($"  \tIpv4Ranges: ");
        permission.Ipv4Ranges.ForEach(range =>
        { Console.WriteLine($"  \t{range.CidrIp} "); });

        Console.WriteLine($"  \n\tIpv6Ranges:");
        permission.Ipv6Ranges.ForEach(range =>
        { Console.WriteLine($"  \t{range.CidrIpv6} "); });

        Console.WriteLine($"  \n\tPrefixListIds: ");
        permission.PrefixListIds.ForEach(id => Console.WriteLine($"  \t{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.WriteLine($"    \tIpv4Ranges: ");
        permission.Ipv4Ranges.ForEach(range =>
        { Console.WriteLine($"    {range.CidrIp} "); });

        Console.WriteLine($"    \n\tIpv6Ranges:");
        permission.Ipv6Ranges.ForEach(range =>
        { Console.WriteLine($"    {range.CidrIpv6} "); });

        Console.WriteLine($"    \n\tPrefixListIds: ");
        permission.PrefixListIds.ForEach(id => Console.WriteLine($"    {id.Id} "));

        Console.WriteLine($"    \n\tTo Port: {permission.ToPort}");
    });
}

```

- Para API obter detalhes, consulte [DescribeSecurityGroup](#) em AWS SDK for .NET API Referência.

Bash

AWS CLI com script Bash

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```

#####
# 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 getopt 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 getopt "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
}

```

As funções utilitárias usadas neste exemplo.

```

#####
# 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
}

```

- Para API obter detalhes, consulte [DescribeSecurityGroups](#) na Referência de AWS CLI Comandos.

C++

SDK para C++

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```

//! Describe all Amazon Elastic Compute Cloud (Amazon EC2) security groups, or a
specific group.

```

```
/*!
 \param groupID: A group ID, ignored if empty.
 \param clientConfiguration: AWS client configuration.
 \return bool: Function succeeded.
 */
bool AwsDoc::EC2::describeSecurityGroups(const Aws::String &groupID,
                                         const Aws::Client::ClientConfiguration
&clientConfiguration) {
    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);
        }

        Aws::EC2::Model::DescribeSecurityGroupsOutcome 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());

return true;
}
```

- Para API obter detalhes, consulte [DescribeSecurityGroup](#) em AWS SDK for C++ API Referência.

CLI

AWS CLI

Exemplo 1: descrever um grupo de segurança

O exemplo `describe-security-groups` a seguir descreve o grupo de segurança especificado.

```
aws ec2 describe-security-groups \
  --group-ids sg-903004f8
```

Saída:

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

Exemplo 2: descrever grupos de segurança com regras específicas

O `describe-security-groups` exemplo a seguir usa filtros para definir o escopo dos resultados para grupos de segurança que têm uma regra que permite SSH tráfego (porta 22) e uma regra que permite tráfego de todos os endereços (`0.0.0.0/0`). O exemplo usa o parâmetro `--query` para exibir somente os nomes dos grupos de segurança. Os grupos de segurança devem corresponder a todos os filtros para serem retornados nos resultados; no entanto, uma única regra não precisa corresponder a todos os filtros. Por exemplo, a saída retorna um grupo de segurança com uma regra que permite SSH tráfego de um endereço IP específico e outra regra que permite HTTP tráfego de todos os endereços.

```
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
```

Saída:

```
default
my-security-group
web-servers
launch-wizard-1
```

Exemplo 3: descrever grupos de segurança com base em tags

O exemplo a seguir `describe-security-groups` usa filtros para definir o escopo dos resultados para grupos de segurança que incluem `test` no nome do grupo de segurança e contêm a tag `Test=To-delete`. O exemplo usa o `--query` parâmetro para exibir somente os nomes e os grupos IDs de segurança.

```
aws ec2 describe-security-groups \
  --filters Name=group-name,Values=*test* Name=tag:Test,Values=To-delete \
  --query "SecurityGroups[*].{Name:GroupName, ID:GroupId}"
```

Saída:

```
[
  {
    "Name": "testfornewinstance",
    "ID": "sg-33bb22aa"
  },
```

```
{
  "Name": "newgroupptest",
  "ID": "sg-1a2b3c4d"
}
]
```

Para obter mais exemplos de uso de filtros de tags, consulte Como [trabalhar com tags](#) no Guia EC2 do usuário da Amazon.

- Para API obter detalhes, consulte [DescribeSecurityGroups](#) na Referência de AWS CLI Comandos.

Java

SDK para Java 2.x

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
/**
 * Asynchronously describes the security groups for the specified group ID.
 *
 * @param groupName the name of the security group to describe
 * @return a {@link CompletableFuture} that represents the asynchronous
operation
 *       of describing the security groups. The future will complete with a
 *       {@link DescribeSecurityGroupsResponse} object that contains the
 *       security group information.
 */
public CompletableFuture<String> describeSecurityGroupArnByNameAsync(String
groupName) {
    DescribeSecurityGroupsRequest request =
DescribeSecurityGroupsRequest.builder()
        .groupNames(groupName)
        .build();

    DescribeSecurityGroupsPublisher paginator =
getAsyncClient().describeSecurityGroupsPaginator(request);
```

```

        AtomicReference<String> groupIdRef = new AtomicReference<>();
        return paginator.subscribe(response -> {
            response.securityGroups().stream()
                .filter(securityGroup ->
securityGroup.groupName().equals(groupIdName))
                .findFirst()
                .ifPresent(securityGroup ->
groupIdRef.set(securityGroup.groupId()));
        }).thenApply(v -> {
            String groupId = groupIdRef.get();
            if (groupId == null) {
                throw new RuntimeException("No security group found with the
name: " + groupIdName);
            }
            return groupId;
        }).exceptionally(ex -> {
            logger.info("Failed to describe security group: " + ex.getMessage());
            throw new RuntimeException("Failed to describe security group", ex);
        });
    }
}

```

- Para API obter detalhes, consulte [DescribeSecurityGroup](#) em AWS SDK for Java 2.x API Referência.

JavaScript

SDK para JavaScript (v3)

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```

import { DescribeSecurityGroupsCommand, EC2Client } from "@aws-sdk/client-ec2";

/**
 * Describes the specified security groups or all of your security groups.
 * @param {{ groupIds: string[] }} options
 */

```

```
export const main = async ({ groupIds = [] }) => {
  const client = new EC2Client({});
  const command = new DescribeSecurityGroupsCommand({
    GroupIds: groupIds,
  });

  try {
    const { SecurityGroups } = await client.send(command);
    const sgList = SecurityGroups.map(
      (sg) => `• ${sg.GroupName} (${sg.GroupId}): ${sg.Description}`,
    ).join("\n");
    if (sgList.length) {
      console.log(`Security groups:\n${sgList}`);
    } else {
      console.log("No security groups found.");
    }
  } catch (caught) {
    if (caught instanceof Error && caught.name === "InvalidGroupId.Malformed") {
      console.warn(`${caught.message}. Please provide a valid GroupId.`);
    } else if (
      caught instanceof Error &&
      caught.name === "InvalidGroup.NotFound"
    ) {
      console.warn(caught.message);
    } else {
      throw caught;
    }
  }
};
```

- Para API obter detalhes, consulte [DescribeSecurityGroup](#) em AWS SDK for JavaScript API Referência.

Kotlin

SDK para Kotlin

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
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}")
        }
    }
}
```

- Para API obter detalhes, consulte a [DescribeSecurityGroups](#) referência AWS SDK do Kotlin API.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo descreve o grupo de segurança especificado para VPC a. Ao trabalhar com grupos de segurança pertencentes a um, VPC você deve usar o ID do grupo de segurança (- GroupId parâmetro), não o nome (- GroupName parâmetro), para referenciar o grupo.

```
Get-EC2SecurityGroup -GroupId sg-12345678
```

Saída:

```
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
```

Exemplo 2: Este exemplo descreve o grupo de segurança especificado para EC2 -Classic. Ao trabalhar com grupos de segurança para EC2 -Classic, você pode usar o nome do grupo (-GroupName parâmetro) ou o ID do grupo (-GroupId parâmetro) para referenciar o grupo de segurança.

```
Get-EC2SecurityGroup -GroupName my-security-group
```

Saída:

```
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          :
```

Exemplo 3: Este exemplo recupera todos os grupos de segurança do vpc-0fc1ff23456b789eb

```
Get-EC2SecurityGroup -Filter @{Name="vpc-id";Values="vpc-0fc1ff23456b789eb"}
```

- Para API obter detalhes, consulte [DescribeSecurityGroups](#) em Referência de AWS Tools for PowerShell cmdlet.

Python

SDK para Python (Boto3)

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

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

    def __init__(self, ec2_client: boto3.client, security_group: Optional[str] =
    None):
        """
        Initializes the SecurityGroupWrapper with an EC2 client and an optional
        security group ID.

        :param ec2_client: A Boto3 Amazon EC2 client. This client provides low-
        level
                           access to AWS EC2 services.
        :param security_group: The ID of a security group to manage. This is a
        high-level identifier
                           that represents the security group.
        """
        self.ec2_client = ec2_client
        self.security_group = security_group

    @classmethod
    def from_client(cls) -> "SecurityGroupWrapper":
        """
        Creates a SecurityGroupWrapper instance with a default EC2 client.

        :return: An instance of SecurityGroupWrapper initialized with the default
        EC2 client.
        """
        ec2_client = boto3.client("ec2")
        return cls(ec2_client)
```



```
def describe(self, security_group_id: Optional[str] = None) -> bool:
    """
    Displays information about the specified security group or all security
    groups if no ID is provided.

    :param security_group_id: The ID of the security group to describe.
        If None, an open search is performed to
    describe all security groups.
    :returns: True if the description is successful.
    :raises ClientError: If there is an error describing the security
    group(s), such as an invalid security group ID.
    """
    try:
        paginator = self.ec2_client.get_paginator("describe_security_groups")

        if security_group_id is None:
            # If no ID is provided, return all security groups.
            page_iterator = paginator.paginate()
        else:
            page_iterator = paginator.paginate(GroupIds=[security_group_id])

        for page in page_iterator:
            for security_group in page["SecurityGroups"]:
                print(f"Security group: {security_group['GroupName']}")
                print(f"\tID: {security_group['GroupId']}")
                print(f"\tVPC: {security_group['VpcId']}")
                if security_group["IpPermissions"]:
                    print("Inbound permissions:")
                    pp(security_group["IpPermissions"])

        return True
    except ClientError as err:
        logger.error("Failed to describe security group(s).")
        if err.response["Error"]["Code"] == "InvalidGroup.NotFound":
            logger.error(
                f"Security group {security_group_id} does not exist "
                f"because the specified security group ID was not found."
            )
        raise
```

- Para API obter detalhes, consulte a [DescribeSecurityGroups](#) Referência AWS SDK do Python (Boto3). API

Rust

SDK para Rust

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
async fn show_security_groups(client: &aws_sdk_ec2::Client, group_ids:
Vec<String>) {
    let response = client
        .describe_security_groups()
        .set_group_ids(Some(group_ids))
        .send()
        .await;

    match response {
        Ok(output) => {
            for group in output.security_groups() {
                println!(
                    "Found Security Group {} ({}), vpc id {} and description {}",
                    group.group_name().unwrap_or("unknown"),
                    group.group_id().unwrap_or("id-unknown"),
                    group.vpc_id().unwrap_or("vpcid-unknown"),
                    group.description().unwrap_or("(none)")
                );
            }
        }
        Err(err) => {
            let err = err.into_service_error();
            let meta = err.meta();
            let message = meta.message().unwrap_or("unknown");
            let code = meta.code().unwrap_or("unknown");
            eprintln!("Error listing EC2 Security Groups: ({code}) {message}");
        }
    }
}
```

```
}

```

- Para API obter detalhes, consulte [DescribeSecurityGroups](#) AWS SDK API referência do Rust.

SAP ABAP

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```

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.

```

- Para API obter detalhes, consulte [DescribeSecurityGroups AWS SDK para SAP ABAP API referência](#).

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use `DescribeSnapshotAttribute` com um CLI

Os exemplos de código a seguir mostram como usar o `DescribeSnapshotAttribute`.

CLI

AWS CLI

Para descrever os atributos de um instantâneo

O `describe-snapshot-attribute` exemplo a seguir lista as contas com as quais um snapshot é compartilhado.

```
aws ec2 describe-snapshot-attribute \  
  --snapshot-id snap-01234567890abcdef \  
  --attribute createVolumePermission
```

Saída:

```
{  
  "SnapshotId": "snap-01234567890abcdef",  
  "CreateVolumePermissions": [  
    {  
      "UserId": "123456789012"  
    }  
  ]  
}
```

Para obter mais informações, consulte [Compartilhe um EBS snapshot da Amazon no Guia](#) do usuário do Amazon Elastic Compute Cloud.

- Para API obter detalhes, consulte [DescribeSnapshotAttribute](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo descreve o atributo especificado do instantâneo especificado.

```
Get-EC2SnapshotAttribute -SnapshotId snap-12345678 -Attribute ProductCodes
```

Saída:

```

CreateVolumePermissions    ProductCodes    SnapshotId
-----
{}                          {}               snap-12345678

```

Exemplo 2: Este exemplo descreve o atributo especificado do instantâneo especificado.

```

(Get-EC2SnapshotAttribute -SnapshotId snap-12345678 -Attribute
 CreateVolumePermission).CreateVolumePermissions

```

Saída:

```

Group    UserId
-----
all

```

- Para API obter detalhes, consulte [DescribeSnapshotAttribute](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **DescribeSnapshots** com um AWS SDK ou CLI

Os exemplos de código a seguir mostram como usar o DescribeSnapshots.

CLI**AWS CLI**

Exemplo 1: descrever um snapshot

O exemplo describe-snapshots a seguir descreve o snapshot especificado.

```

aws ec2 describe-snapshots \
  --snapshot-ids snap-1234567890abcdef0

```

Saída:

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

Para obter mais informações, consulte [Amazon EBS snapshots](#) no Guia do EC2 usuário da Amazon.

Exemplo 2: descrever snapshots com base em filtros

O `describe-snapshots` exemplo a seguir usa filtros para definir o escopo dos resultados para instantâneos pertencentes à sua AWS conta que estão no `pending` estado. O exemplo usa o `--query` parâmetro para exibir somente o instantâneo IDs e a hora em que o instantâneo foi iniciado.

```
aws ec2 describe-snapshots \
  --owner-ids self \
  --filters Name=status,Values=pending \
  --query "Snapshots[*].{ID:SnapshotId,Time:StartTime}"
```

Saída:

```
[
  {
```

```
    "ID": "snap-1234567890abcdef0",
    "Time": "2019-08-04T12:48:18.000Z"
  },
  {
    "ID": "snap-066877671789bd71b",
    "Time": "2019-08-04T02:45:16.000Z"
  },
  ...
]
```

O exemplo `describe-snapshots` a seguir usa filtros para definir o escopo dos resultados para snapshots criados no volume especificado. O exemplo usa o `--query` parâmetro para exibir somente o `instantâneoIDs`.

```
aws ec2 describe-snapshots \
  --filters Name=volume-id,Values=049df61146c4d7901 \
  --query "Snapshots[*].[SnapshotId]" \
  --output text
```

Saída:

```
snap-1234567890abcdef0
snap-08637175a712c3fb9
...
```

Para obter exemplos adicionais de uso de filtros, consulte [Listar e filtrar seus recursos](#) no Guia do EC2 usuário da Amazon.

Exemplo 3: descrever snapshots com base em tags

O exemplo `describe-snapshots` a seguir usa filtros de tag para definir o escopo dos resultados para snapshots que tenham a tag `Stack=Prod`.

```
aws ec2 describe-snapshots \
  --filters Name=tag:Stack,Values=prod
```

Para obter um exemplo da saída de `describe-snapshots`, consulte o Exemplo 1.

Para obter mais exemplos de uso de filtros de tags, consulte Como [trabalhar com tags](#) no Guia EC2 do usuário da Amazon.

Exemplo 4: descrever snapshots com base na idade

O `describe-snapshots` exemplo a seguir usa JMESPath expressões para descrever todos os instantâneos criados pela sua AWS conta antes da data especificada. Ele exibe somente o `instantâneoIDs`.

```
aws ec2 describe-snapshots \  
  --owner-ids 012345678910 \  
  --query "Snapshots[?(StartTime<='2020-03-31')].[SnapshotId]"
```

Para obter exemplos adicionais de uso de filtros, consulte [Listar e filtrar seus recursos](#) no Guia do EC2 usuário da Amazon.

Exemplo 5: visualizar somente snapshots arquivados

O exemplo `describe-snapshots` a seguir lista apenas os snapshots armazenados no nível de arquivamento.

```
aws ec2 describe-snapshots \  
  --filters "Name=storage-tier,Values=archive"
```

Saída:

```
{  
  "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": []  
    },  
  ]  
}
```


Para obter mais informações, consulte [View archived snapshots](#) no Guia do usuário do Amazon Elastic Compute Cloud.

- Para API obter detalhes, consulte [DescribeSnapshots](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo descreve o instantâneo especificado.

```
Get-EC2Snapshot -SnapshotId snap-12345678
```

Saída:

```
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
```

Exemplo 2: Este exemplo descreve os instantâneos que têm uma tag "Nome".

```
Get-EC2Snapshot | ? { $_.Tags.Count -gt 0 -and $_.Tags.Key -eq "Name" }
```

Exemplo 3: Este exemplo descreve os instantâneos que têm uma tag 'Nome' com o valor 'TestValue'.

```
Get-EC2Snapshot | ? { $_.Tags.Count -gt 0 -and $_.Tags.Key -eq "Name" -and  
$_.Tags.Value -eq "TestValue" }
```

Exemplo 4: Este exemplo descreve todos os seus instantâneos.

```
Get-EC2Snapshot -Owner self
```

- Para API obter detalhes, consulte [DescribeSnapshots](#) em Referência de AWS Tools for PowerShell cmdlet.

Rust

SDK para Rust

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

Mostra o estado de um snapshot.

```
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(())
}
```

- Para API obter detalhes, consulte [DescribeSnapshots](#) a AWS SDK API referência do Rust.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **DescribeSpotDatafeedSubscription** com um CLI

Os exemplos de código a seguir mostram como usar o `DescribeSpotDatafeedSubscription`.

CLI

AWS CLI

Para descrever a assinatura do feed de dados da Spot Instance para uma conta

Este exemplo de comando descreve o feed de dados da conta.

Comando:

```
aws ec2 describe-spot-datafeed-subscription
```

Saída:

```
{
  "SpotDatafeedSubscription": {
    "OwnerId": "123456789012",
    "Prefix": "spotdata",
    "Bucket": "my-s3-bucket",
    "State": "Active"
  }
}
```

- Para API obter detalhes, consulte [DescribeSpotDatafeedSubscription](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Esse exemplo descreve o feed de dados da sua instância spot.

```
Get-EC2SpotDatafeedSubscription
```

Saída:

```
Bucket   : my-s3-bucket
Fault    :
OwnerId  : 123456789012
Prefix   : spotdata
State    : Active
```

- Para API obter detalhes, consulte [DescribeSpotDatafeedSubscription](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use `DescribeSpotFleetInstances` com um CLI

Os exemplos de código a seguir mostram como usar o `DescribeSpotFleetInstances`.

CLI

AWS CLI

Para descrever as instâncias spot associadas a uma frota spot

Este exemplo de comando lista as instâncias spot associadas à frota spot especificada.

Comando:

```
aws ec2 describe-spot-fleet-instances --spot-fleet-request-id sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE
```

Saída:

```
{
  "ActiveInstances": [
    {
      "InstanceId": "i-1234567890abcdef0",
      "InstanceType": "m3.medium",
      "SpotInstanceRequestId": "sir-08b93456"
    },
    ...
  ],
  "SpotFleetRequestId": "sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE"
}
```

- Para API obter detalhes, consulte [DescribeSpotFleetInstances](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo descreve as instâncias associadas à solicitação de frota spot especificada.

```
Get-EC2SpotFleetInstance -SpotFleetRequestId sfr-73fbd2ce-
aa30-494c-8788-1cee4EXAMPLE
```

Saída:

InstanceId	InstanceType	SpotInstanceRequestId
i-f089262a	c3.large	sir-12345678
i-7e8b24a4	c3.large	sir-87654321

- Para API obter detalhes, consulte [DescribeSpotFleetInstances](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use `DescribeSpotFleetRequestHistory` com um CLI

Os exemplos de código a seguir mostram como usar o `DescribeSpotFleetRequestHistory`.

CLI

AWS CLI

Para descrever o histórico da frota Spot

Esse exemplo de comando retorna o histórico da frota spot especificada a partir do horário especificado.

Comando:

```
aws ec2 describe-spot-fleet-request-history --spot-fleet-request-id sfr-73fbd2ce-
aa30-494c-8788-1cee4EXAMPLE --start-time 2015-05-26T00:00:00Z
```

O exemplo de saída a seguir mostra os lançamentos bem-sucedidos de duas instâncias spot para a frota spot.

Saída:

```
{
  "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+TfG1WxWWB/Vr5dk5d4LfdgA/DRAHUrYgxzrEXAMPLE=",
  "StartTime": "2015-05-26T00:00:00Z"
}
```

- Para API obter detalhes, consulte [DescribeSpotFleetRequestHistory](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo descreve o histórico da solicitação de frota spot especificada.

```
Get-EC2SpotFleetRequestHistory -SpotFleetRequestId sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE -StartTime 2015-12-26T00:00:00Z
```

Saída:

```
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
```

Saída:

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

- Para API obter detalhes, consulte [DescribeSpotFleetRequestHistory](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **DescribeSpotFleetRequests** com um CLI

Os exemplos de código a seguir mostram como usar o `DescribeSpotFleetRequests`.

CLI

AWS CLI

Para descrever suas solicitações de frota Spot

Este exemplo descreve todas as suas solicitações de frota spot.

Comando:

```
aws ec2 describe-spot-fleet-requests
```

Saída:

```
{
  "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"
}
]
}

```

Para descrever uma solicitação de frota spot

Este exemplo descreve a solicitação de frota spot especificada.

Comando:

```
aws ec2 describe-spot-fleet-requests --spot-fleet-request-ids sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE
```

Saída:

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

- Para API obter detalhes, consulte [DescribeSpotFleetRequests](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo descreve a solicitação de frota spot especificada.

```
Get-EC2SpotFleetRequest -SpotFleetRequestId sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE | format-list
```

Saída:

```
ConfigData           : Amazon.EC2.Model.SpotFleetRequestConfigData
CreateTime           : 12/26/2015 8:23:33 AM
SpotFleetRequestId  : sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE
SpotFleetRequestState : active
```

Exemplo 2: Este exemplo descreve todas as suas solicitações de frota spot.

```
Get-EC2SpotFleetRequest
```

- Para API obter detalhes, consulte [DescribeSpotFleetRequests](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **DescribeSpotInstanceRequests** com um CLI

Os exemplos de código a seguir mostram como usar o `DescribeSpotInstanceRequests`.

CLI

AWS CLI

Exemplo 1: Para descrever uma solicitação de instância spot

O `describe-spot-instance-requests` exemplo a seguir descreve a solicitação de instância spot especificada.

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

Saída:

```
{  
  "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"
}
]
}

```

Exemplo 2: Para descrever solicitações de Instância Spot com base em filtros

O `describe-spot-instance-requests` exemplo a seguir usa filtros para definir o escopo dos resultados para solicitações de instância spot com o tipo de instância especificado na zona de disponibilidade especificada. O exemplo usa o `--query` parâmetro para exibir somente a instânciasIDs.

```

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

```

Saída:

```
i-057750d42936e468a
i-001efd250faaa6ffa
i-027552a73f021f3bd
...
```

Para obter exemplos adicionais de uso de filtros, consulte [Listar e filtrar seus recursos no Guia do usuário do Amazon Elastic Compute Cloud](#).

Exemplo 3: Para descrever solicitações de Instância Spot com base em tags

O `describe-spot-instance-requests` exemplo a seguir usa filtros de tag para definir o escopo dos resultados para solicitações de Instância Spot que têm a `tagcost-center=cc123`.

```
aws ec2 describe-spot-instance-requests \
  --filters Name=tag:cost-center,Values=cc123
```

Para obter um exemplo da saída de `describe-spot-instance-requests`, consulte o Exemplo 1.

Para obter mais exemplos de uso de filtros de tags, consulte Como [trabalhar com tags](#) no Guia EC2 do usuário da Amazon.

- Para API obter detalhes, consulte [DescribeSpotInstanceRequests](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo descreve a solicitação de instância spot especificada.

```
Get-EC2SpotInstanceRequest -SpotInstanceRequestId sir-12345678
```

Saída:

```
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
```

Exemplo 2: Este exemplo descreve todas as suas solicitações de instância spot.

```
Get-EC2SpotInstanceRequest
```

- Para API obter detalhes, consulte [DescribeSpotInstanceRequests](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **DescribeSpotPriceHistory** com um CLI

Os exemplos de código a seguir mostram como usar o `DescribeSpotPriceHistory`.

CLI

AWS CLI

Para descrever o histórico de preços à vista

Esse exemplo de comando retorna o histórico de preços spot para instâncias m1.xlarge em um determinado dia de janeiro.

Comando:


```
aws ec2 describe-spot-price-history --instance-types m1.xlarge --start-time 2014-01-06T07:08:09 --end-time 2014-01-06T08:09:10
```

Saída:

```
{
  "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"
    },
    ...
  ]
}
```

Para descrever o histórico de preços spot para Linux/Amazon UNIX VPC

Esse comando de exemplo retorna o histórico de preços spot para VPC instâncias m1.xlarge, UNIX Linux/Amazon de um determinado dia de janeiro.

Comando:

```
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
```

Saída:

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

- Para API obter detalhes, consulte [DescribeSpotPriceHistory](#) na Referência de AWS CLI Comandos.

PowerShell**Ferramentas para PowerShell**

Exemplo 1: Esse exemplo obtém as últimas 10 entradas no histórico de preços spot para o tipo de instância e a zona de disponibilidade especificados. Observe que o valor especificado para o AvailabilityZone parâmetro - deve ser válido para o valor da região fornecido ao parâmetro -Region do cmdlet (não mostrado no exemplo) ou definido como padrão no shell. Este exemplo de comando pressupõe que uma região padrão de 'us-west-2' tenha sido definida no ambiente.

```
Get-EC2SpotPriceHistory -InstanceType c3.large -AvailabilityZone us-west-2a -
MaxResult 10
```

Saída:

```
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
...
```

- Para API obter detalhes, consulte [DescribeSpotPriceHistory](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **DescribeSubnets** com um AWS SDK ou CLI

Os exemplos de código a seguir mostram como usar o `DescribeSubnets`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação no contexto no seguinte exemplo de código:

- [Criar e gerenciar um serviço resiliente](#)

.NET

AWS SDK for .NET

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
/// <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)
{
    try
    {
        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;
    }
    catch (AmazonEC2Exception ec2Exception)
```

```
    {
        if (ec2Exception.ErrorCode == "InvalidVpcID.NotFound")
        {
            _logger.LogError(ec2Exception, $"The specified VPC ID {vpcId}
does not exist.");
        }

        throw;
    }
    catch (Exception ex)
    {
        _logger.LogError(ex, $"An error occurred while describing the
subnets.: {ex.Message}");
        throw;
    }
}
```

- Para API obter detalhes, consulte [DescribeSubnets](#) em AWS SDK for .NET API Referência.

CLI

AWS CLI

Exemplo 1: descrever todas as suas sub-redes

O exemplo `describe-subnets` a seguir mostra os detalhes das suas sub-redes.

```
aws ec2 describe-subnets
```

Saída:

```
{
  "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": 4096,
    "CidrBlock": "172.31.80.0/20",
    "DefaultForAz": true,
    "MapPublicIpOnLaunch": true,
    "MapCustomerOwnedIpOnLaunch": false,
    "State": "available",
    "SubnetId": "subnet-8EXAMPLE",
    "VpcId": "vpc-3EXAMPLE",
    "OwnerId": "111122223333",
    "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
    }
}

```

```

    }
  }
]
}

```

Para obter mais informações, consulte [Trabalho com sub-redes VPCs e sub-redes](#) no Guia do AWS VPC usuário.

Exemplo 2: Para descrever as sub-redes de uma rede específica VPC

O `describe-subnets` exemplo a seguir usa um filtro para recuperar detalhes das sub-redes especificadas. VPC

```

aws ec2 describe-subnets \
  --filters "Name=vpc-id,Values=vpc-3EXAMPLE"

```

Saída:

```

{
  "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
        }
    ]
}
```

Para obter mais informações, consulte [Trabalho com sub-redes VPCs e sub-redes](#) no Guia do AWS VPC usuário.

Exemplo 3: descrever as sub-redes com uma tag específica

O `describe-subnets` exemplo a seguir usa um filtro para recuperar os detalhes dessas sub-redes com a tag `CostCenter=123` e o `--query` parâmetro para exibir a sub-rede IDs das sub-redes com essa tag.

```
aws ec2 describe-subnets \
  --filters "Name=tag:CostCenter,Values=123" \
  --query "Subnets[*].SubnetId" \
  --output text
```

Saída:

```
subnet-0987a87c8b37348ef
subnet-02a95061c45f372ee
subnet-03f720e7de2788d73
```

Para obter mais informações, consulte [Trabalho com sub-redes VPCs e sub-redes no Guia VPC](#) do usuário da Amazon.

- Para API obter detalhes, consulte [DescribeSubnets](#) na Referência de AWS CLI Comandos.

JavaScript

SDK para JavaScript (v3)

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
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"] },
    ],
  }),
);
```

- Para API obter detalhes, consulte [DescribeSubnets](#) em AWS SDK for JavaScript API Referência.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo descreve a sub-rede especificada.

```
Get-EC2Subnet -SubnetId subnet-1a2b3c4d
```

Saída:

```
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
```

Exemplo 2: Este exemplo descreve todas as suas sub-redes.

```
Get-EC2Subnet
```

- Para API obter detalhes, consulte [DescribeSubnets](#) em Referência de AWS Tools for PowerShell cmdlet.

Python

SDK para Python (Boto3)

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
class AutoScalingWrapper:
    """
    Encapsulates Amazon EC2 Auto Scaling and EC2 management actions.
    """

    def __init__(
        self,
        resource_prefix: str,
        inst_type: str,
        ami_param: str,
        autoscaling_client: boto3.client,
        ec2_client: boto3.client,
        ssm_client: boto3.client,
        iam_client: boto3.client,
    ):
        """
        Initializes the AutoScaler class with the necessary parameters.

        :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
sts_client = boto3.client("sts")
self.account_id = sts_client.get_caller_identity()["Account"]

self.key_pair_name = f"{resource_prefix}-key-pair"
self.launch_template_name = f"{resource_prefix}-template-"
self.group_name = f"{resource_prefix}-group"

# Happy path
self.instance_policy_name = f"{resource_prefix}-pol"
self.instance_role_name = f"{resource_prefix}-role"
self.instance_profile_name = f"{resource_prefix}-prof"

# Failure mode
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"

def get_subnets(self, vpc_id: str, zones: List[str] = None) -> List[Dict[str,
Any]]:
    """
    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.
    """
    # Ensure that 'zones' is a list, even if None is passed
    if zones is None:

```

```
zones = []
try:
    paginator = self.ec2_client.get_paginator("describe_subnets")
    page_iterator = paginator.paginate(
        Filters=[
            {"Name": "vpc-id", "Values": [vpc_id]},
            {"Name": "availability-zone", "Values": zones},
            {"Name": "default-for-az", "Values": ["true"]},
        ]
    )

    subnets = []
    for page in page_iterator:
        subnets.extend(page["Subnets"])

    log.info("Found %s subnets for the specified zones.", len(subnets))
    return subnets
except ClientError as err:
    log.error(
        f"Failed to retrieve subnets for VPC '{vpc_id}' in zones
{zones}."
    )
    error_code = err.response["Error"]["Code"]
    if error_code == "InvalidVpcID.NotFound":
        log.error(
            "The specified VPC ID does not exist. "
            "Please check the VPC ID and try again."
        )
    # Add more error-specific handling as needed
    log.error(f"Full error:\n\t{err}")
```

- Para API obter detalhes, consulte a [DescribeSubnets](#) Referência AWS SDK do Python (Boto3). API

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **DescribeTags** com um CLI

Os exemplos de código a seguir mostram como usar o DescribeTags.

CLI

AWS CLI

Exemplo 1: Para descrever todas as tags de um único recurso

O describe-tags exemplo a seguir descreve as tags da instância especificada.

```
aws ec2 describe-tags \  
  --filters "Name=resource-id,Values=i-1234567890abcdef8"
```

Saída:

```
{  
  "Tags": [  
    {  
      "ResourceType": "instance",  
      "ResourceId": "i-1234567890abcdef8",  
      "Value": "Test",  
      "Key": "Stack"  
    },  
    {  
      "ResourceType": "instance",  
      "ResourceId": "i-1234567890abcdef8",  
      "Value": "Beta Server",  
      "Key": "Name"  
    }  
  ]  
}
```

Exemplo 2: Para descrever todas as tags de um tipo de recurso

O describe-tags exemplo a seguir descreve as tags dos seus volumes.

```
aws ec2 describe-tags \  
  --filters "Name=resource-type,Values=volume"
```

Saída:

```
{
  "Tags": [
    {
      "ResourceType": "volume",
      "ResourceId": "vol-1234567890abcdef0",
      "Value": "Project1",
      "Key": "Purpose"
    },
    {
      "ResourceType": "volume",
      "ResourceId": "vol-049df61146c4d7901",
      "Value": "Logs",
      "Key": "Purpose"
    }
  ]
}
```

Exemplo 3: Para descrever todas as suas tags

O `describe-tags` exemplo a seguir descreve as tags de todos os seus recursos.

```
aws ec2 describe-tags
```

Exemplo 4: Para descrever as tags de seus recursos com base em uma chave de tag

O `describe-tags` exemplo a seguir descreve as tags dos seus recursos que têm uma tag com a chave `Stack`.

```
aws ec2 describe-tags \
  --filters Name=key,Values=Stack
```

Saída:

```
{
  "Tags": [
    {
      "ResourceType": "volume",
      "ResourceId": "vol-027552a73f021f3b",
      "Value": "Production",
      "Key": "Stack"
    },
  ],
}
```

```

    {
      "ResourceType": "instance",
      "ResourceId": "i-1234567890abcdef8",
      "Value": "Test",
      "Key": "Stack"
    }
  ]
}

```

Exemplo 5: Para descrever as tags de seus recursos com base na chave e no valor da tag

O `describe-tags` exemplo a seguir descreve as tags dos seus recursos que têm a `tagStack=Test`.

```

aws ec2 describe-tags \
  --filters Name=key,Values=Stack Name=value,Values=Test

```

Saída:

```

{
  "Tags": [
    {
      "ResourceType": "image",
      "ResourceId": "ami-3ac336533f021f3bd",
      "Value": "Test",
      "Key": "Stack"
    },
    {
      "ResourceType": "instance",
      "ResourceId": "i-1234567890abcdef8",
      "Value": "Test",
      "Key": "Stack"
    }
  ]
}

```

O `describe-tags` exemplo a seguir usa uma sintaxe alternativa para descrever recursos com a `tagStack=Test`.

```

aws ec2 describe-tags \
  --filters "Name=tag:Stack,Values=Test"

```

O `describe-tags` exemplo a seguir descreve as tags de todas as suas instâncias que têm uma tag com a chave `Purpose` e sem valor.

```
aws ec2 describe-tags \
  --filters "Name=resource-
  type,Values=instance" "Name=key,Values=Purpose" "Name=value,Values="
```

Saída:

```
{
  "Tags": [
    {
      "ResourceType": "instance",
      "ResourceId": "i-1234567890abcdef5",
      "Value": null,
      "Key": "Purpose"
    }
  ]
}
```

- Para API obter detalhes, consulte [DescribeTags](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo busca as tags para o tipo de recurso 'image'

```
Get-EC2Tag -Filter @{Name="resource-type";Values="image"}
```

Saída:

Key	ResourceId	ResourceType	Value
---	-----	-----	-----
Name	ami-0a123b4ccb567a8ea	image	Win7-Imported
auto-delete	ami-0a123b4ccb567a8ea	image	never

Exemplo 2: Este exemplo busca todas as tags de todos os recursos e as agrupa por tipo de recurso


```
Get-EC2Tag | Group-Object resourcetype
```

Saída:

```
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}
```

Exemplo 3: Este exemplo exibe todos os recursos com a tag 'autodelete' com o valor 'no' para a região em questão

```
Get-EC2Tag -Region eu-west-1 -Filter @{Name="tag:auto-delete";Values="no"}
```

Saída:

```
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

```

Exemplo 4: este exemplo obtém todos os recursos com a tag “exclusão automática” com valor “nenhum” e filtros adicionais no próximo canal para analisar somente os tipos de recursos de “instância” e, eventualmente, cria a tag “ThisInstance” para cada recurso da instância, com o valor sendo o próprio ID da instância

```

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}}

```

Exemplo 5: este exemplo busca tags para todos os recursos da instância, bem como para as chaves “Name”, e as exibe em formato de tabela

```

Get-EC2Tag -Filter @{{Name="resource-
type";Values="instance"},@{{Name="key";Values="Name"}} | Select-Object ResourceId,
@{{Name="Name-Tag";Expression={$PSItem.Value}} | Format-Table -AutoSize

```

Saída:

```

ResourceId          Name-Tag
-----
i-012e3cb4df567e1aa jump1
i-01c23a45d6fc7a89f repro-3

```

- Para API obter detalhes, consulte [DescribeTags](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **DescribeVolumeAttribute** com um CLI

Os exemplos de código a seguir mostram como usar o `DescribeVolumeAttribute`.

CLI

AWS CLI

Para descrever um atributo de volume

Este exemplo de comando descreve o `autoEnableIo` atributo do volume com o ID `vol-049df61146c4d7901`.

Comando:

```
aws ec2 describe-volume-attribute --volume-id vol-049df61146c4d7901 --  
attribute autoEnableIO
```

Saída:

```
{  
  "AutoEnableIO": {  
    "Value": false  
  },  
  "VolumeId": "vol-049df61146c4d7901"  
}
```

- Para API obter detalhes, consulte [DescribeVolumeAttribute](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo descreve o atributo especificado do volume especificado.

```
Get-EC2VolumeAttribute -VolumeId vol-12345678 -Attribute AutoEnableIO
```

Saída:

AutoEnableIO	ProductCodes	VolumeId
-----	-----	-----
False	{}	vol-12345678

- Para API obter detalhes, consulte [DescribeVolumeAttribute](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use `DescribeVolumeStatus` com um CLI

Os exemplos de código a seguir mostram como usar o `DescribeVolumeStatus`.

CLI

AWS CLI

Para descrever o status de um único volume

Este exemplo de comando descreve o status do volume `vol-1234567890abcdef0`.

Comando:

```
aws ec2 describe-volume-status --volume-ids vol-1234567890abcdef0
```

Saída:

```
{
  "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": []
    }
]
}

```

Para descrever o status dos volumes danificados

Este exemplo de comando descreve o status de todos os volumes danificados. Neste exemplo de saída, não há volumes comprometidos.

Comando:

```
aws ec2 describe-volume-status --filters Name=volume-
status.status,Values=impaired
```

Saída:

```
{
  "VolumeStatuses": []
}
```

Se você tiver um volume com falha na verificação de status (o status está comprometido), consulte Como trabalhar com um volume prejudicado no Guia EC2 do usuário da Amazon.

- Para API obter detalhes, consulte [DescribeVolumeStatus](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo descreve o status do volume especificado.

```
Get-EC2VolumeStatus -VolumeId vol-12345678
```

Saída:

```
Actions           : {}
AvailabilityZone  : us-west-2a
Events           : {}
```

```
VolumeId      : vol-12345678
VolumeStatus  : Amazon.EC2.Model.VolumeStatusInfo
```

```
(Get-EC2VolumeStatus -VolumeId vol-12345678).VolumeStatus
```

Saída:

```
Details      Status
-----
{io-enabled, io-performance} ok
```

```
(Get-EC2VolumeStatus -VolumeId vol-12345678).VolumeStatus.Details
```

Saída:

```
Name      Status
----
io-enabled passed
io-performance not-applicable
```

- Para API obter detalhes, consulte [DescribeVolumeStatus](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **DescribeVolumes** com um CLI

Os exemplos de código a seguir mostram como usar o `DescribeVolumes`.

CLI

AWS CLI

Exemplo 1: Para descrever um volume

O `describe-volumes` exemplo a seguir descreve os volumes especificados na região atual.

```
aws ec2 describe-volumes \
```

```
--volume-ids vol-049df61146c4d7901 vol-1234567890abcdef0
```

Saída:

```
{
  "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
    }
  ]
}
```

Exemplo 2: Para descrever volumes que estão anexados a uma instância específica

O `describe-volumes` exemplo a seguir descreve todos os volumes anexados à instância especificada e definidos para serem excluídos quando a instância for encerrada.

```
aws ec2 describe-volumes \  
  --region us-east-1 \  
  --filters Name=attachment.instance-  
id,Values=i-1234567890abcdef0 Name=attachment.delete-on-termination,Values=true
```

Para obter um exemplo da saída de `describe-volumes`, consulte o Exemplo 1.

Exemplo 3: Para descrever os volumes disponíveis em uma zona de disponibilidade específica

O `describe-volumes` exemplo a seguir descreve todos os volumes que têm um status de `available` e estão na zona de disponibilidade especificada.

```
aws ec2 describe-volumes \  
  --filters Name=status,Values=available Name=availability-zone,Values=us-  
east-1a
```

Para obter um exemplo da saída de `describe-volumes`, consulte o Exemplo 1.

Exemplo 4: Para descrever volumes com base em tags

O `describe-volumes` exemplo a seguir descreve todos os volumes que têm a chave de tag `Name` e um valor que começa com `Test`. A saída é então filtrada com uma consulta que exibe somente as tags e IDs os volumes.

```
aws ec2 describe-volumes \  
  --filters Name=tag:Name,Values=Test* \  
  --query "Volumes[*].{ID:VolumeId,Tag:Tags}"
```

Saída:

```
[  
  {  
    "Tag": [  
      {  
        "Value": "Test2",  
        "Key": "Name"      }  
    ]  
  }  
]
```



```
    }
  ],
  "ID": "vol-1234567890abcdef0"
},
{
  "Tag": [
    {
      "Value": "Test1",
      "Key": "Name"
    }
  ],
  "ID": "vol-049df61146c4d7901"
}
]
```

Para obter mais exemplos de uso de filtros de tags, consulte Como [trabalhar com tags](#) no Guia EC2 do usuário da Amazon.

- Para API obter detalhes, consulte [DescribeVolumes](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo descreve o EBS volume especificado.

```
Get-EC2Volume -VolumeId vol-12345678
```

Saída:

```
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
```

Exemplo 2: Este exemplo descreve seus EBS volumes que têm o status 'disponível'.

```
Get-EC2Volume -Filter @{ Name="status"; Values="available" }
```

Saída:

```
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
...
```

Exemplo 3: Este exemplo descreve todos os seus EBS volumes.

```
Get-EC2Volume
```

- Para API obter detalhes, consulte [DescribeVolumes](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **DescribeVpcAttribute** com um CLI

Os exemplos de código a seguir mostram como usar o `DescribeVpcAttribute`.

CLI

AWS CLI

Para descrever o `enableDnsSupport` atributo

Este exemplo descreve o `enableDnsSupport` atributo. Esse atributo indica se a DNS resolução está habilitada para VPC o. Se esse atributo for `true`, o DNS servidor da Amazon resolverá os DNS nomes de host de suas instâncias para os endereços IP correspondentes; caso contrário, isso não acontece.

Comando:

```
aws ec2 describe-vpc-attribute --vpc-id vpc-a01106c2 --attribute enableDnsSupport
```

Saída:

```
{
  "VpcId": "vpc-a01106c2",
  "EnableDnsSupport": {
    "Value": true
  }
}
```

Para descrever o `enableDnsHostnames` atributo

Este exemplo descreve o `enableDnsHostnames` atributo. Esse atributo indica se as instâncias foram executadas no VPC get DNS hostnames. Se esse atributo for `true`, instâncias em VPC get DNS hostnames; caso contrário, não o farão.

Comando:

```
aws ec2 describe-vpc-attribute --vpc-id vpc-a01106c2 --
attribute enableDnsHostnames
```

Saída:

```
{
  "VpcId": "vpc-a01106c2",
  "EnableDnsHostnames": {
    "Value": true
  }
}
```

- Para API obter detalhes, consulte [DescribeVpcAttribute](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo descreve o atributo enableDnsSupport ".

```
Get-EC2VpcAttribute -VpcId vpc-12345678 -Attribute enableDnsSupport
```

Saída:

```
EnableDnsSupport  
-----  
True
```

Exemplo 2: Este exemplo descreve o atributo enableDnsHostnames ".

```
Get-EC2VpcAttribute -VpcId vpc-12345678 -Attribute enableDnsHostnames
```

Saída:

```
EnableDnsHostnames  
-----  
True
```

- Para API obter detalhes, consulte [DescribeVpcAttribute](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **DescribeVpcClassicLink** com um CLI

Os exemplos de código a seguir mostram como usar o DescribeVpcClassicLink.

CLI

AWS CLI

Para descrever o ClassicLink status do seu VPCs

Este exemplo lista o ClassicLink status de vpc-88888888.

Comando:

```
aws ec2 describe-vpc-classic-link --vpc-id vpc-88888888
```

Saída:

```
{
  "Vpcs": [
    {
      "ClassicLinkEnabled": true,
      "VpcId": "vpc-88888888",
      "Tags": [
        {
          "Value": "classiclinkvpc",
          "Key": "Name"
        }
      ]
    }
  ]
}
```

Este exemplo lista somente os VPCs que estão habilitados para o Classiclink (o valor do filtro de `is-classic-link-enabled` está definido como). `true`

Comando:

```
aws ec2 describe-vpc-classic-link --filter "Name=is-classic-link-enabled,Values=true"
```

- Para API obter detalhes, consulte [DescribeVpcClassicLink](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: O exemplo acima retorna todos os VPCs com seu ClassicLinkEnabled estado para a região

```
Get-EC2VpcClassicLink -Region eu-west-1
```

Saída:

```
ClassicLinkEnabled Tags    VpcId
-----
False              {Name} vpc-0fc1ff23f45b678eb
False              {}      vpc-01e23c4a5d6db78e9
False              {Name} vpc-0123456b078b9d01f
False              {}      vpc-12cf3b4f
False              {Name} vpc-0b12d3456a7e8901d
```

- Para API obter detalhes, consulte [DescribeVpcClassicLink](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **DescribeVpcClassicLinkDnsSupport** com um CLI

Os exemplos de código a seguir mostram como usar o `DescribeVpcClassicLinkDnsSupport`.

CLI

AWS CLI

Para descrever ClassicLink DNS o suporte para seu VPCs

Este exemplo descreve o status de ClassicLink DNS suporte de todos os seus VPCs.

Comando:

```
aws ec2 describe-vpc-classic-link-dns-support
```

Saída:

```
{
  "Vpcs": [
    {
```

```

    "VpcId": "vpc-88888888",
    "ClassicLinkDnsSupported": true
  },
  {
    "VpcId": "vpc-1a2b3c4d",
    "ClassicLinkDnsSupported": false
  }
]
}

```

- Para API obter detalhes, consulte [DescribeVpcClassicLinkDnsSupport](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo descreve o status de ClassicLink DNS suporte da VPCs região eu-west-1

```
Get-EC2VpcClassicLinkDnsSupport -VpcId vpc-0b12d3456a7e8910d -Region eu-west-1
```

Saída:

```

ClassicLinkDnsSupported VpcId
-----
False                   vpc-0b12d3456a7e8910d
False                   vpc-12cf3b4f

```

- Para API obter detalhes, consulte [DescribeVpcClassicLinkDnsSupport](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **DescribeVpcEndpointServices** com um CLI

Os exemplos de código a seguir mostram como usar o `DescribeVpcEndpointServices`.

CLI

AWS CLI

Exemplo 1: Para descrever todos os serviços de VPC endpoint

O exemplo "describe-vpc-endpoint-services" a seguir lista todos os serviços de VPC endpoint de uma AWS região.

```
aws ec2 describe-vpc-endpoint-services
```

Saída:

```
{
  "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"
  ]
}

```

Para obter mais informações, consulte [Exibir nomes AWS de serviços disponíveis](#) no Guia do usuário do AWS PrivateLink.

Exemplo 2: Para descrever os detalhes sobre um serviço de endpoint

O exemplo "describe-vpc-endpoint-services" a seguir lista os detalhes do serviço de endpoint da interface Amazon S3

```

aws ec2 describe-vpc-endpoint-services \
  --filter "Name=service-type,Values=Interface" Name=service-
  name,Values=com.amazonaws.us-east-1.s3

```

Saída:

```

{
  "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"
]
}

```

Para obter mais informações, consulte [Exibir nomes AWS de serviços disponíveis](#) no Guia do usuário do AWS PrivateLink.

- Para API obter detalhes, consulte [DescribeVpcEndpointServices](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo descreve o serviço de EC2 VPC endpoint com o filtro fornecido, neste caso com.amazonaws.eu-west-1.ecs. Além disso, ele também expande a ServiceDetails propriedade e exibe os detalhes

```

Get-EC2VpcEndpointService -Region eu-west-1 -MaxResult 5 -Filter @{Name="service-name";Values="com.amazonaws.eu-west-1.ecs"} | Select-Object -ExpandProperty ServiceDetails

```

Saída:

```

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

```

Exemplo 2: Este exemplo recupera todos os serviços do EC2 VPC Endpoint e retorna o “ssm” ServiceNames correspondente

```
Get-EC2VpcEndpointService -Region eu-west-1 | Select-Object -ExpandProperty
  Servicenames | Where-Object { -match "ssm"}
```

Saída:

```
com.amazonaws.eu-west-1.ssm
com.amazonaws.eu-west-1.ssmmessages
```

- Para API obter detalhes, consulte [DescribeVpcEndpointServices](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **DescribeVpcEndpoints** com um CLI

Os exemplos de código a seguir mostram como usar o DescribeVpcEndpoints.

CLI

AWS CLI

Para descrever seus VPC endpoints

O describe-vpc-endpoints exemplo a seguir exibe detalhes de todos os seus VPC endpoints.

```
aws ec2 describe-vpc-endpoints
```

Saída:

```
{
  "VpcEndpoints": [
    {
      "PolicyDocument": "{\"Version\":\"2008-10-17\",\"Statement\":
[{\\"Effect\\":\\"Allow\\",\\"Principal\\":\\"*\\",\\"Action\\":\\"*\\",\\"Resource\\":\\"*
\\"}]}",
```



```

        "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-111122223333aabbcc",
    "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"
}
]
}

```

Para obter mais informações, consulte [VPCendpoints](#) no Guia do VPC usuário da Amazon.

- Para API obter detalhes, consulte [DescribeVpcEndpoints](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo descreve um ou mais dos seus VPC endpoints para a região eu-west-1. Em seguida, ele canaliza a saída para o próximo comando, que seleciona a VpcEndpointId propriedade e retorna o VPC ID da matriz como matriz de string

```
Get-EC2VpcEndpoint -Region eu-west-1 | Select-Object -ExpandProperty
  VpcEndpointId
```

Saída:

```
vpce-01a2ab3f4f5cc6f7d
vpce-01d2b345a6787890b
vpce-0012e34d567890e12
vpce-0c123db4567890123
```

Exemplo 2: Este exemplo descreve todos os endpoints vpc da região eu-west-1 e seleciona VpcEndpointId,, e as propriedades para apresentá-los em formato Vpclid tabular ServiceName PrivateDnsEnabled

```
Get-EC2VpcEndpoint -Region eu-west-1 | Select-Object VpcEndpointId, VpcId,
  ServiceName, PrivateDnsEnabled | Format-Table -AutoSize
```

Saída:

VpcEndpointId	VpcId	ServiceName
vpce-02a2ab2f2f2cc2f2d	vpc-0fc6ff46f65b039eb	com.amazonaws.eu-west-1.ssm
PrivateDnsEnabled		
True		
vpce-01d1b111a1114561b	vpc-0fc6ff46f65b039eb	com.amazonaws.eu-west-1.ec2
PrivateDnsEnabled		
True		
vpce-0011e23d45167e838	vpc-0fc6ff46f65b039eb	com.amazonaws.eu-west-1.ec2messages
PrivateDnsEnabled		
True		
vpce-0c123db4567890123	vpc-0fc6ff46f65b039eb	com.amazonaws.eu-west-1.ssmessages
PrivateDnsEnabled		
True		

Exemplo 3: Este exemplo exporta o documento de política do VPC Endpoint `vpce-01a2ab3f4f5cc6f7d` em um arquivo json

```
Get-EC2VpcEndpoint -Region eu-west-1 -VpcEndpointId vpce-01a2ab3f4f5cc6f7d |  
Select-Object -expand PolicyDocument | Out-File vpce_policyDocument.json
```

- Para API obter detalhes, consulte [DescribeVpcEndpoints](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **DescribeVpcs** com um AWS SDK ou CLI

Os exemplos de código a seguir mostram como usar o `DescribeVpcs`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação no contexto no seguinte exemplo de código:

- [Criar e gerenciar um serviço resiliente](#)

.NET

AWS SDK for .NET

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
/// <summary>  
/// Get the default VPC for the account.  
/// </summary>  
/// <returns>The default VPC object.</returns>  
public async Task<Vpc> GetDefaultVpc()  
{  
    try  
    {
```



```
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];
}
catch (AmazonEC2Exception ec2Exception)
{
    if (ec2Exception.ErrorCode == "UnauthorizedOperation")
    {
        _logger.LogError(ec2Exception, $"You do not have the necessary
permissions to describe VPCs.");
    }

    throw;
}
catch (Exception ex)
{
    _logger.LogError(ex, $"An error occurred while describing the vpcs.:
{ex.Message}");
    throw;
}
}
```

- Para API obter detalhes, consulte [DescribeVpcs](#) em AWS SDK for .NET API Referência.

CLI

AWS CLI

Exemplo 1: Para descrever todas as suas VPCs

O `describe-vpcs` exemplo a seguir recupera detalhes sobre seu VPCs.

```
aws ec2 describe-vpcs
```

Saída:

```
{
  "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"
            }
        ]
    }
]
}

```

Exemplo 2: Para descrever um determinado VPC

O `describe-vpcs` exemplo a seguir recupera detalhes do especificado VPC.

```

aws ec2 describe-vpcs \
  --vpc-ids vpc-06e4ab6c6cEXAMPLE

```

Saída:

```

{
  "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"
        }
      ]
    }
  ]
}

```

```

    ]
  }
]
}

```

- Para API obter detalhes, consulte [DescribeVpcs](#) na Referência de AWS CLI Comandos.

JavaScript

SDK para JavaScript (v3)

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```

const client = new EC2Client({});
const { Vpcs } = await client.send(
  new DescribeVpcsCommand({
    Filters: [{ Name: "is-default", Values: ["true"] }],
  }),
);

```

- Para API obter detalhes, consulte [DescribeVpcs](#) em AWS SDK for JavaScript API Referência.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo descreve o especificado VPC.

```
Get-EC2Vpc -VpcId vpc-12345678
```

Saída:

```

CidrBlock      : 10.0.0.0/16
DhcpOptionsId  : dopt-1a2b3c4d

```

```
InstanceTenancy : default
IsDefault       : False
State          : available
Tags           : {Name}
VpcId         : vpc-12345678
```

Exemplo 2: Este exemplo descreve o padrão VPC (só pode haver um por região). Se sua conta suportar EC2 -Classic nessa região, não há padrãoVPC.

```
Get-EC2Vpc -Filter @{{Name="isDefault"; Values="true"}}
```

Saída:

```
CidrBlock       : 172.31.0.0/16
DhcpOptionsId  : dopt-12345678
InstanceTenancy : default
IsDefault      : True
State         : available
Tags          : {}
VpcId        : vpc-45678901
```

Exemplo 3: Este exemplo descreve os VPCs que correspondem ao filtro especificado (ou seja, têm um CIDR que corresponde ao valor '10.0.0.0/16' e estão no estado 'disponível').

```
Get-EC2Vpc -Filter @{{Name="cidr";
  Values="10.0.0.0/16"},@{{Name="state";Values="available"}}
```

Exemplo 4: Este exemplo descreve todos os seusVPCs.

```
Get-EC2Vpc
```

- Para API obter detalhes, consulte [DescribeVpcs](#) em Referência de AWS Tools for PowerShell cmdlet.

Python

SDK para Python (Boto3)

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
class AutoScalingWrapper:
    """
    Encapsulates Amazon EC2 Auto Scaling and EC2 management actions.
    """

    def __init__(
        self,
        resource_prefix: str,
        inst_type: str,
        ami_param: str,
        autoscaling_client: boto3.client,
        ec2_client: boto3.client,
        ssm_client: boto3.client,
        iam_client: boto3.client,
    ):
        """
        Initializes the AutoScaler class with the necessary parameters.

        :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
sts_client = boto3.client("sts")
self.account_id = sts_client.get_caller_identity()["Account"]

self.key_pair_name = f"{resource_prefix}-key-pair"
self.launch_template_name = f"{resource_prefix}-template-"
self.group_name = f"{resource_prefix}-group"

# Happy path
self.instance_policy_name = f"{resource_prefix}-pol"
self.instance_role_name = f"{resource_prefix}-role"
self.instance_profile_name = f"{resource_prefix}-prof"

# Failure mode
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"

def get_default_vpc(self) -> Dict[str, Any]:
    """
    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:
        error_code = err.response["Error"]["Code"]
        log.error("Failed to retrieve the default VPC.")
        if error_code == "UnauthorizedOperation":
            log.error(
                "You do not have the necessary permissions to describe VPCs.
                "
                "Ensure that your AWS IAM user or role has the correct
                permissions."
            )
        elif error_code == "InvalidParameterValue":
            log.error(
                "One or more parameters are invalid. Check the request
                parameters."
```

```
    )

    log.error(f"Full error:\n\t{err}")
else:
    if "Vpcs" in response and response["Vpcs"]:
        log.info(f"Retrieved default VPC: {response['Vpcs'][0]
['VpcId']}")
        return response["Vpcs"][0]
    else:
        pass
```

- Para API obter detalhes, consulte a [DescribeVpcs](#) Referência AWS SDK do Python (Boto3).
API

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **DescribeVpnConnections** com um CLI

Os exemplos de código a seguir mostram como usar o `DescribeVpnConnections`.

CLI

AWS CLI

Exemplo 1: Para descrever suas VPN conexões

O `describe-vpn-connections` exemplo a seguir descreve todas as suas Site-to-Site VPN conexões.

```
aws ec2 describe-vpn-connections
```

Saída:

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

```

Para obter mais informações, consulte [Como AWS Site-to-Site VPN funciona](#) no Guia AWS Site-to-Site VPN do usuário.

Exemplo 2: Para descrever suas VPN conexões disponíveis

O `describe-vpn-connections` exemplo a seguir descreve suas Site-to-Site VPN conexões com um estado `deavailable`.

```
aws ec2 describe-vpn-connections \  
  --filters "Name=state,Values=available"
```

Para obter mais informações, consulte [Como AWS Site-to-Site VPN funciona](#) no Guia AWS Site-to-Site VPN do usuário.

- Para API obter detalhes, consulte [DescribeVpnConnections](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo descreve a VPN conexão especificada.

```
Get-EC2VpnConnection -VpnConnectionId vpn-12345678
```

Saída:

```
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
```

Exemplo 2: Este exemplo descreve qualquer VPN conexão cujo estado esteja pendente ou disponível.

```
$filter = New-Object Amazon.EC2.Model.Filter  
$filter.Name = "state"  
$filter.Values = @( "pending", "available" )
```

```
Get-EC2VpnConnection -Filter $filter
```

Exemplo 3: Este exemplo descreve todas as suas VPN conexões.

```
Get-EC2VpnConnection
```

- Para API obter detalhes, consulte [DescribeVpnConnections](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **DescribeVpnGateways** com um CLI

Os exemplos de código a seguir mostram como usar o DescribeVpnGateways.

CLI

AWS CLI

Para descrever seus gateways privados virtuais

Este exemplo descreve seus gateways privados virtuais.

Comando:

```
aws ec2 describe-vpn-gateways
```

Saída:

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

```

- Para API obter detalhes, consulte [DescribeVpnGateways](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo descreve o gateway privado virtual especificado.

```
Get-EC2VpnGateway -VpnGatewayId vgw-1a2b3c4d
```

Saída:

```

AvailabilityZone :
State            : available
Tags            : {}
Type            : ipsec.1
VpcAttachments  : {vpc-12345678}
VpnGatewayId    : vgw-1a2b3c4d

```

Exemplo 2: Este exemplo descreve qualquer gateway privado virtual cujo estado esteja pendente ou disponível.

```
$filter = New-Object Amazon.EC2.Model.Filter
```

```
$filter.Name = "state"
$filter.Values = @( "pending", "available" )

Get-EC2VpnGateway -Filter $filter
```

Exemplo 3: Este exemplo descreve todos os seus gateways privados virtuais.

```
Get-EC2VpnGateway
```

- Para API obter detalhes, consulte [DescribeVpnGateways](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **DetachInternetGateway** com um CLI

Os exemplos de código a seguir mostram como usar o DetachInternetGateway.

CLI

AWS CLI

Para separar um gateway de internet do seu VPC

O detach-internet-gateway exemplo a seguir separa o gateway de internet especificado do específicoVPC.

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

Este comando não produz saída.

Para obter mais informações, consulte [Gateways de Internet](#) no Guia do VPC usuário da Amazon.

- Para API obter detalhes, consulte [DetachInternetGateway](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo separa o gateway de Internet especificado do especificadoVPC.

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

- Para API obter detalhes, consulte [DetachInternetGateway](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **DetachNetworkInterface** com um CLI

Os exemplos de código a seguir mostram como usar o `DetachNetworkInterface`.

CLI

AWS CLI

Para separar uma interface de rede da sua instância

Este exemplo separa a interface de rede especificada da instância especificada. Se o comando for bem-sucedido, nenhuma saída será retornada.

Comando:

```
aws ec2 detach-network-interface --attachment-id eni-attach-66c4350a
```

- Para API obter detalhes, consulte [DetachNetworkInterface](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Esse exemplo remove o anexo especificado entre uma interface de rede e uma instância.

```
Dismount-EC2NetworkInterface -AttachmentId eni-attach-1a2b3c4d -Force
```

- Para API obter detalhes, consulte [DetachNetworkInterface](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **DetachVolume** com um CLI

Os exemplos de código a seguir mostram como usar o DetachVolume.

CLI

AWS CLI

Para separar um volume de uma instância

Este exemplo de comando separa o volume (vol-049df61146c4d7901) da instância à qual ele está conectado.

Comando:

```
aws ec2 detach-volume --volume-id vol-1234567890abcdef0
```

Saída:

```
{
  "AttachTime": "2014-02-27T19:23:06.000Z",
  "InstanceId": "i-1234567890abcdef0",
  "VolumeId": "vol-049df61146c4d7901",
  "State": "detaching",
  "Device": "/dev/sdb"
}
```

- Para API obter detalhes, consulte [DetachVolume](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo separa o volume especificado.

```
Dismount-EC2Volume -VolumeId vol-12345678
```

Saída:

```
AttachTime      : 12/22/2015 1:53:58 AM
DeleteOnTermination : False
Device          : /dev/sdh
InstanceId      : i-1a2b3c4d
State          : detaching
VolumeId       : vol-12345678
```

Exemplo 2: Você também pode especificar o ID da instância e o nome do dispositivo para garantir que você esteja desanexando o volume correto.

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

- Para API obter detalhes, consulte [DetachVolume](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **DetachVpnGateway** com um CLI

Os exemplos de código a seguir mostram como usar o DetachVpnGateway.

CLI

AWS CLI

Para separar um gateway privado virtual do seu VPC

Este exemplo separa o gateway privado virtual especificado do especificadoVPC. Se o comando for bem-sucedido, nenhuma saída será retornada.

Comando:

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

- Para API obter detalhes, consulte [DetachVpnGateway](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo separa o gateway privado virtual especificado do especificado VPC.

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

- Para API obter detalhes, consulte [DetachVpnGateway](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **DisableVgwRoutePropagation** com um CLI

Os exemplos de código a seguir mostram como usar o `DisableVgwRoutePropagation`.

CLI

AWS CLI

Para desativar a propagação da rota

Este exemplo impede que o gateway privado virtual especificado propague rotas estáticas para a tabela de rotas especificada. Se o comando for bem-sucedido, nenhuma saída será retornada.

Comando:

```
aws ec2 disable-vgw-route-propagation --route-table-id rtb-22574640 --gateway-id vgw-9a4cacf3
```

- Para API obter detalhes, consulte [DisableVgwRoutePropagation](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo desativa a propagação automática VGW de rotas para a tabela de roteamento especificada.

```
Disable-EC2VgwRoutePropagation -RouteTableId rtb-12345678 -GatewayId vgw-1a2b3c4d
```

- Para API obter detalhes, consulte [DisableVgwRoutePropagation](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **DisableVpcClassicLink** com um CLI

Os exemplos de código a seguir mostram como usar o `DisableVpcClassicLink`.

CLI

AWS CLI

Para desativar ClassicLink para um VPC

Este exemplo desativa ClassicLink para `vpc-88888888`.

Comando:

```
aws ec2 disable-vpc-classic-link --vpc-id vpc-88888888
```

Saída:

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

- Para API obter detalhes, consulte [DisableVpcClassicLink](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo é desativado EC2VpcClassicLink para o vpc-01e23c4a5d6db78e9. Ele retorna Verdadeiro ou Falso

```
Disable-EC2VpcClassicLink -VpcId vpc-01e23c4a5d6db78e9
```

- Para API obter detalhes, consulte [DisableVpcClassicLink](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **DisableVpcClassicLinkDnsSupport** com um CLI

Os exemplos de código a seguir mostram como usar o `DisableVpcClassicLinkDnsSupport`.

CLI

AWS CLI

Para desativar o ClassicLink DNS suporte para um VPC

Este exemplo desativa o ClassicLink DNS suporte para `vpc-88888888`.

Comando:

```
aws ec2 disable-vpc-classic-link-dns-support --vpc-id vpc-88888888
```

Saída:

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

```
}
```

- Para API obter detalhes, consulte [DisableVpcClassicLinkDnsSupport](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo desativa o ClassicLink DNS suporte para o vpc-0b12d3456a7e8910d

```
Disable-EC2VpcClassicLinkDnsSupport -VpcId vpc-0b12d3456a7e8910d
```

- Para API obter detalhes, consulte [DisableVpcClassicLinkDnsSupport](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **DisassociateAddress** com um AWS SDK ou CLI

Os exemplos de código a seguir mostram como usar o `DisassociateAddress`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação no contexto no seguinte exemplo de código:

- [Aprender os conceitos básicos](#)

.NET

AWS SDK for .NET

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
/// <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)
{
    try
    {
        var response = await _amazonEC2.DisassociateAddressAsync(
            new DisassociateAddressRequest { AssociationId =
associationId });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }
    catch (AmazonEC2Exception ec2Exception)
    {
        if (ec2Exception.ErrorCode == "InvalidAssociationID.NotFound")
        {
            _logger.LogError(
                $"AssociationId is invalid, unable to disassociate address.
{ec2Exception.Message}");
        }

        return false;
    }
    catch (Exception ex)
    {
        _logger.LogError(
            $"An error occurred while disassociating the Elastic IP.:
{ex.Message}");
        return false;
    }
}
```

- Para API obter detalhes, consulte [DisassociateAddress](#) em AWS SDK for .NET API Referência.

Bash

AWS CLI com script Bash

 Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
#####
# 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
}

```

As funções utilitárias usadas neste exemplo.

```

#####
# 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
}

```

- Para API obter detalhes, consulte [DisassociateAddress](#) na Referência de AWS CLI Comandos.

CLI

AWS CLI

Para desassociar um endereço IP elástico em EC2 -Classic

Este exemplo dissocia um endereço IP elástico de uma instância em EC2 -Classic. Se o comando for bem-sucedido, nenhuma saída será retornada.

Comando:

```
aws ec2 disassociate-address --public-ip 198.51.100.0
```

Para desassociar um endereço IP elástico em EC2 - VPC

Este exemplo dissocia um endereço IP elástico de uma instância em um VPC. Se o comando for bem-sucedido, nenhuma saída será retornada.

Comando:

```
aws ec2 disassociate-address --association-id eipassoc-2bebb745
```

- Para API obter detalhes, consulte [DisassociateAddress](#) na Referência de AWS CLI Comandos.

Java

SDK para Java 2.x

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
/**
 * Disassociates an Elastic IP address from an instance asynchronously.
 *
 * @param associationId The ID of the association you want to disassociate.
 * @return a {@link CompletableFuture} representing the asynchronous
 * operation of disassociating the address. The
 *         {@link CompletableFuture} will complete with a {@link
 * DisassociateAddressResponse} when the operation is
 *         finished.
 * @throws RuntimeException if the disassociation of the address fails.
 */
```

```
public CompletableFuture<DisassociateAddressResponse>
disassociateAddressAsync(String associationId) {
    Ec2AsyncClient ec2 = getAsyncClient();
    DisassociateAddressRequest addressRequest =
DisassociateAddressRequest.builder()
        .associationId(associationId)
        .build();

    // Disassociate the address asynchronously.
    CompletableFuture<DisassociateAddressResponse> response =
ec2.disassociateAddress(addressRequest);
    response.whenComplete((resp, ex) -> {
        if (ex != null) {
            throw new RuntimeException("Failed to disassociate address", ex);
        }
    });

    return response;
}
```

- Para API obter detalhes, consulte [DisassociateAddress](#) em AWS SDK for Java 2.x API Referência.

JavaScript

SDK para JavaScript (v3)

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
import { DisassociateAddressCommand, EC2Client } from "@aws-sdk/client-ec2";

/**
 * Disassociate an Elastic IP address from an instance.
 * @param {{ associationId: string }} options
 */
export const main = async ({ associationId }) => {
```

```
const client = new EC2Client({});
const command = new DisassociateAddressCommand({
  // You can also use PublicIp, but that is for EC2 classic which is being
  retired.
  AssociationId: associationId,
});

try {
  await client.send(command);
  console.log("Successfully disassociated address");
} catch (caught) {
  if (
    caught instanceof Error &&
    caught.name === "InvalidAssociationID.NotFound"
  ) {
    console.warn(`${caught.message}.`);
  } else {
    throw caught;
  }
}
};
```

- Para API obter detalhes, consulte [DisassociateAddress](#) em AWS SDK for JavaScript API Referência.

Kotlin

SDK para Kotlin

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
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!")
}
}
```

- Para API obter detalhes, consulte a [DisassociateAddress](#) referência AWS SDK do Kotlin API.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo dissocia o endereço IP elástico especificado da instância especificada em um VPC

```
Unregister-EC2Address -AssociationId eipassoc-12345678
```

Exemplo 2: Este exemplo dissocia o endereço IP elástico especificado da instância especificada em EC2 -Classic.

```
Unregister-EC2Address -PublicIp 203.0.113.17
```

- Para API obter detalhes, consulte [DisassociateAddress](#) em Referência de AWS Tools for PowerShell cmdlet.

Python

SDK para Python (Boto3)

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

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

```

class ElasticIp:
    """Represents an Elastic IP and its associated instance."""

    def __init__(
        self, allocation_id: str, public_ip: str, instance_id: Optional[str]
= None
    ) -> None:
        """
        Initializes the ElasticIp object.

        :param allocation_id: The allocation ID of the Elastic IP.
        :param public_ip: The public IP address of the Elastic IP.
        :param instance_id: The ID of the associated EC2 instance, if any.
        """
        self.allocation_id = allocation_id
        self.public_ip = public_ip
        self.instance_id = instance_id

    def __init__(self, ec2_client: Any) -> None:
        """
        Initializes the ElasticIpWrapper with an EC2 client.

        :param ec2_client: A Boto3 Amazon EC2 client. This client provides low-
level
                           access to AWS EC2 services.
        """
        self.ec2_client = ec2_client
        self.elastic_ips: List[ElasticIpWrapper.ElasticIp] = []

    @classmethod
    def from_client(cls) -> "ElasticIpWrapper":
        """
        Creates an ElasticIpWrapper instance with a default EC2 client.

        :return: An instance of ElasticIpWrapper initialized with the default EC2
client.
        """
        ec2_client = boto3.client("ec2")
        return cls(ec2_client)

    def disassociate(self, allocation_id: str) -> None:
        """

```

```
    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.

    :param allocation_id: The allocation ID of the Elastic IP to
disassociate.
    :raises ClientError: If the disassociation fails, such as when the
association ID is not found.
    """
    elastic_ip = self.get_elastic_ip_by_allocation(self.elastic_ips,
allocation_id)
    if elastic_ip is None or elastic_ip.instance_id is None:
        logger.info(
            f"No association found for Elastic IP with allocation ID
{allocation_id}."
        )
        return

    try:
        # Retrieve the association ID before disassociating
        response =
self.ec2_client.describe_addresses(AllocationIds=[allocation_id])
        association_id = response["Addresses"][0].get("AssociationId")

        if association_id:

self.ec2_client.disassociate_address(AssociationId=association_id)
            elastic_ip.instance_id = None # Remove the instance association
        else:
            logger.info(
                f"No Association ID found for Elastic IP with allocation ID
{allocation_id}."
            )

    except ClientError as err:
        if err.response["Error"]["Code"] == "InvalidAssociationID.NotFound":
            logger.error(
                f"Failed to disassociate Elastic IP {allocation_id} "
                "because the specified association ID for the Elastic IP
address was not found. "
                "Verify the association ID and ensure the Elastic IP is
currently associated with a "
                "resource before attempting to disassociate it."
            )
```

```
raise
```

- Para API obter detalhes, consulte a [DisassociateAddress](#) Referência AWS SDK do Python (Boto3). API

Rust

SDK para Rust

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
pub async fn disassociate_ip_address(&self, association_id: &str) ->
Result<(), EC2Error> {
    self.client
        .disassociate_address()
        .association_id(association_id)
        .send()
        .await?;
    Ok(())
}
```

- Para API obter detalhes, consulte [DisassociateAddress](#) AWS SDK API referência do Rust.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **DisassociateRouteTable** com um CLI

Os exemplos de código a seguir mostram como usar o `DisassociateRouteTable`.

CLI

AWS CLI

Para desassociar uma tabela de rotas

Este exemplo dissocia a tabela de rotas especificada da sub-rede especificada. Se o comando for bem-sucedido, nenhuma saída será retornada.

Comando:

```
aws ec2 disassociate-route-table --association-id rtbassoc-781d0d1a
```

- Para API obter detalhes, consulte [DisassociateRouteTable](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Esse exemplo remove a associação especificada entre uma tabela de rotas e uma sub-rede.

```
Unregister-EC2RouteTable -AssociationId rtbassoc-1a2b3c4d
```

- Para API obter detalhes, consulte [DisassociateRouteTable](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **EnableVgwRoutePropagation** com um CLI

Os exemplos de código a seguir mostram como usar o `EnableVgwRoutePropagation`.

CLI

AWS CLI

Para ativar a propagação da rota

Este exemplo permite que o gateway privado virtual especificado propague rotas estáticas para a tabela de rotas especificada. Se o comando for bem-sucedido, nenhuma saída será retornada.

Comando:

```
aws ec2 enable-vgw-route-propagation --route-table-id rtb-22574640 --gateway-id vgw-9a4cacf3
```

- Para API obter detalhes, consulte [EnableVgwRoutePropagation](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo permite que o especificado VGW propague rotas automaticamente para a tabela de roteamento especificada.

```
Enable-EC2VgwRoutePropagation -RouteTableId rtb-12345678 -GatewayId vgw-1a2b3c4d
```

- Para API obter detalhes, consulte [EnableVgwRoutePropagation](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **EnableVolumeIo** com um CLI

Os exemplos de código a seguir mostram como usar o `EnableVolumeIo`.

CLI

AWS CLI

Para habilitar a E/S para um volume

Este exemplo habilita a E/S no volume `vol-1234567890abcdef0`.

Comando:

```
aws ec2 enable-volume-io --volume-id vol-1234567890abcdef0
```

Saída:

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

- Para API obter detalhes, consulte [EnableVolumeIo](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo habilita operações de E/S para o volume especificado, se as operações de E/S estiverem desativadas.

```
Enable-EC2VolumeIO -VolumeId vol-12345678
```

- Para API obter detalhes, consulte [EnableVolumeIo](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **EnableVpcClassicLink** com um CLI

Os exemplos de código a seguir mostram como usar o `EnableVpcClassicLink`.

CLI

AWS CLI

Para habilitar um VPC para ClassicLink

Este exemplo habilita o vpc-88888888 para ClassicLink

Comando:

```
aws ec2 enable-vpc-classic-link --vpc-id vpc-88888888
```

Saída:

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

- Para API obter detalhes, consulte [EnableVpcClassicLink](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo ativa o VPC vpc-0123456b789b0d12f para ClassicLink

```
Enable-EC2VpcClassicLink -VpcId vpc-0123456b789b0d12f
```

Saída:

```
True
```

- Para API obter detalhes, consulte [EnableVpcClassicLink](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **EnableVpcClassicLinkDnsSupport** com um CLI

Os exemplos de código a seguir mostram como usar o `EnableVpcClassicLinkDnsSupport`.

CLI

AWS CLI

Para habilitar o ClassicLink DNS suporte para um VPC

Este exemplo habilita ClassicLink DNS o suporte paravpc-88888888.

Comando:

```
aws ec2 enable-vpc-classic-link-dns-support --vpc-id vpc-88888888
```

Saída:

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

- Para API obter detalhes, consulte [EnableVpcClassicLinkDnsSupport](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo permite que o vpc-0b12d3456a7e8910d ofereça suporte à resolução de nome de host para DNS ClassicLink

```
Enable-EC2VpcClassicLinkDnsSupport -VpcId vpc-0b12d3456a7e8910d -Region eu-west-1
```

- Para API obter detalhes, consulte [EnableVpcClassicLinkDnsSupport](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **GetConsoleOutput** com um CLI

Os exemplos de código a seguir mostram como usar o GetConsoleOutput.

CLI

AWS CLI

Exemplo 1: Para obter a saída do console

O `get-console-output` exemplo a seguir obtém a saída do console para a instância Linux especificada.

```
aws ec2 get-console-output \  
  --instance-id i-1234567890abcdef0
```

Saída:

```
{  
  "InstanceId": "i-1234567890abcdef0",  
  "Timestamp": "2013-07-25T21:23:53.000Z",  
  "Output": "..."  
}
```

Para obter mais informações, consulte a [saída do console de instância](#) no Amazon EC2 User Guide.

Exemplo 2: Para obter a saída mais recente do console

O `get-console-output` exemplo a seguir obtém a saída mais recente do console para a instância Linux especificada.

```
aws ec2 get-console-output \  
  --instance-id i-1234567890abcdef0 \  
  --latest \  
  --output text
```

Saída:

```
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
```

Para obter mais informações, consulte a [saída do console de instância](#) no Amazon EC2 User Guide.

- Para API obter detalhes, consulte [GetConsoleOutput](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo obtém a saída do console para a instância Linux especificada. A saída do console é codificada.

```
Get-EC2ConsoleOutput -InstanceId i-0e19abcd47c123456
```

Saída:

InstanceId	Output
-----	-----
i-0e194d3c47c123637	WyAgICAwLjAwMDAwMF0gQ29tbW...bGU9dHR5UzAgc2Vs

Exemplo 2: Este exemplo armazena a saída codificada do console em uma variável e, em seguida, a decodifica.

```
$Output_encoded = (Get-EC2ConsoleOutput -InstanceId i-0e19abcd47c123456).Output  
[System.Text.Encoding]::UTF8.GetString([System.Convert]::FromBase64String($Output_encoded))
```

- Para API obter detalhes, consulte [GetConsoleOutput](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use `GetHostReservationPurchasePreview` com um CLI

Os exemplos de código a seguir mostram como usar o `GetHostReservationPurchasePreview`.

CLI

AWS CLI

Para obter uma prévia da compra de uma reserva de anfitrião dedicado

Este exemplo fornece uma prévia dos custos de uma reserva de host dedicado especificada para o host dedicado especificado em sua conta.

Comando:

```
aws ec2 get-host-reservation-purchase-preview --offering-id hro-03f707bf363b6b324
--host-id-set h-013abcd2a00cbd123
```

Saída:

```
{
  "TotalHourlyPrice": "1.499",
  "Purchase": [
    {
      "HourlyPrice": "1.499",
      "InstanceFamily": "m4",
      "PaymentOption": "NoUpfront",
      "HostIdSet": [
        "h-013abcd2a00cbd123"
      ],
      "UpfrontPrice": "0.000",
      "Duration": 31536000
    }
  ],
  "TotalUpfrontPrice": "0.000"
}
```

- Para API obter detalhes, consulte [GetHostReservationPurchasePreview](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo mostra uma compra de reserva com configurações que correspondem às do seu host dedicado h-01e23f4cd567890f1

```
Get-EC2HostReservationPurchasePreview -OfferingId hro-0c1f23456789d0ab -HostIdSet
h-01e23f4cd567890f1
```

Saída:

```
CurrencyCode Purchase TotalHourlyPrice TotalUpfrontPrice
-----
                {}          1.307          0.000
```

- Para API obter detalhes, consulte [GetHostReservationPurchasePreview](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **GetPasswordData** com um AWS SDK ou CLI

Os exemplos de código a seguir mostram como usar o GetPasswordData.

CLI

AWS CLI

Para obter a senha criptografada

Este exemplo obtém a senha criptografada.

Comando:

```
aws ec2 get-password-data --instance-id i-1234567890abcdef0
```

Saída:

```
{
  "InstanceId": "i-1234567890abcdef0",
  "Timestamp": "2013-08-07T22:18:38.000Z",
  "PasswordData": "gSlJFq+VpcZXqy+iktxMF6NyxQ4qCrT4+gaOuN0enX1MmgXPTj7XEXAMPLE
UQ+YeFfb+L1U4C4AKv652Ux1iRB3CPTYP7WmU3TUnhsuBd+p6LVk7T2lKUm160Xbk6WPW1VYYm/TRPB1
e1DQ7PY4an/DgZT4mwcpRfigzhniQgDDe01InvSDcwoUTwNs0Y1S8ouri2W4n5GNlriM3Q0AnNve1Vz/
53TkDtxbNoU606M1gK9zUWSxqEgwwbV2j8c5rP0WCuaMWSF14ziDu4bd7q+4RSyi8NUsVWnKZ4aEZffu
```



```
DPGzKrF5yL1f3etP2L4ZR6CvG7K1hx7VK0QVN32Dajw=="
}
```

Para obter a senha descriptografada

Este exemplo obtém a senha descriptografada.

Comando:

```
aws ec2 get-password-data --instance-id i-1234567890abcdef0 --priv-launch-key C:
\Keys\MyKeyPair.pem
```

Saída:

```
{
  "InstanceId": "i-1234567890abcdef0",
  "Timestamp": "2013-08-30T23:18:05.000Z",
  "PasswordData": "&ViJ652e*u"
}
```

- Para API obter detalhes, consulte [GetPasswordData](#) na Referência de AWS CLI Comandos.

Java

SDK para Java 2.x

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.ec2.Ec2AsyncClient;
import software.amazon.awssdk.services.ec2.model.*;
import java.util.concurrent.CompletableFuture;

/**
 * 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 GetPasswordData {

    public static void main(String[] args) {
        final String usage = ""

            Usage:
                <instanceId>

            Where:
                instanceId - An instance id value that you can obtain from the
AWS Management Console.\s
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }
        String instanceId = args[0];
        Ec2AsyncClient ec2AsyncClient = Ec2AsyncClient.builder()
            .region(Region.US_EAST_1)
            .build();

        try {
            CompletableFuture<Void> future = getPasswordDataAsync(ec2AsyncClient,
instanceId);
            future.join();
        } catch (RuntimeException rte) {
            System.err.println("An exception occurred: " + (rte.getCause() !=
null ? rte.getCause().getMessage() : rte.getMessage()));
        }
    }

    /**
     * Fetches the password data for the specified EC2 instance asynchronously.
     *
     * @param ec2AsyncClient the EC2 asynchronous client to use for the request
     * @param instanceId instanceId the ID of the EC2 instance for which you want
to fetch the password data
    */
}
```

```
    * @return a {@link CompletableFuture} that completes when the password data
    has been fetched
    * @throws RuntimeException if there was a failure in fetching the password
    data
    */
    public static CompletableFuture<Void> getPasswordDataAsync(Ec2AsyncClient
ec2AsyncClient, String instanceId) {
        GetPasswordDataRequest getPasswordDataRequest =
GetPasswordDataRequest.builder()
            .instanceId(instanceId)
            .build();

        CompletableFuture<GetPasswordDataResponse> response =
ec2AsyncClient.getPasswordData(getPasswordDataRequest);
        response.whenComplete((getPasswordDataResponse, ex) -> {
            if (ex != null) {
                throw new RuntimeException("Failed to get password data for
instance: " + instanceId, ex);
            } else if (getPasswordDataResponse == null ||
getPasswordDataResponse.getPasswordData().isEmpty()) {
                throw new RuntimeException("No password data found for instance:
" + instanceId);
            } else {
                String encryptedPasswordData =
getPasswordDataResponse.getPasswordData();
                System.out.println("Encrypted Password Data: " +
encryptedPasswordData);
            }
        });

        return response.thenApply(resp -> null);
    }
}
```

- Para API obter detalhes, consulte [GetPasswordData](#) em AWS SDK for Java 2.x API Referência.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo descriptografa a senha que a Amazon EC2 atribuiu à conta de administrador para a instância especificada do Windows. Quando um arquivo pem foi especificado, a configuração da opção `-Decrypt` é automaticamente assumida.

```
Get-EC2PasswordData -InstanceId i-12345678 -PemFile C:\path\my-key-pair.pem
```

Saída:

```
mYZ(PA9?C)Q
```

Exemplo 2: (PowerShell somente para Windows) inspeciona a instância para determinar o nome do par de chaves usado para iniciar a instância e, em seguida, tenta encontrar os dados do par de chaves correspondente no repositório de configuração do AWS Toolkit for Visual Studio. Se os dados do par de chaves forem encontrados, a senha será descriptografada.

```
Get-EC2PasswordData -InstanceId i-12345678 -Decrypt
```

Saída:

```
mYZ(PA9?C)Q
```

Exemplo 3: retorna os dados da senha criptografada para a instância.

```
Get-EC2PasswordData -InstanceId i-12345678
```

Saída:

```
iVz3BAK/WAXV.....dqt8WeMA==
```

- Para API obter detalhes, consulte [GetPasswordData](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **ImportImage** com um CLI

Os exemplos de código a seguir mostram como usar o `ImportImage`.

CLI

AWS CLI

Para importar um arquivo de imagem da VM como AMI

O `import-image` exemplo a seguir importa o especificado OVA.

```
aws ec2 import-image \  
  --disk-containers Format=ova,UserBucket="{S3Bucket=my-import-bucket,S3Key=vms/  
my-server-vm.ova}"
```

Saída:

```
{  
  "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"  
}
```

- Para API obter detalhes, consulte [ImportImage](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo importa uma imagem de máquina virtual de disco único do bucket do Amazon S3 especificado para a EC2 Amazon com um token de idempotência. O exemplo exige que exista uma função de serviço de importação de VM com o nome padrão 'vmimport', com uma política que permita que a EC2 Amazon acesse o bucket especificado, conforme explicado no tópico Pré-requisitos de importação de VM. Para usar um papel personalizado, especifique o nome do papel usando o **-RoleName** parâmetro.

```
$container = New-Object Amazon.EC2.Model.ImageDiskContainer
$container.Format="VMDK"
$container.UserBucket = New-Object Amazon.EC2.Model.UserBucket
$container.UserBucket.S3Bucket = "amzn-s3-demo-bucket"
$container.UserBucket.S3Key = "Win_2008_Server_Standard_SP2_64-bit-disk1.vmdk"

$params = @{
    "ClientToken"="idempotencyToken"
    "Description"="Windows 2008 Standard Image Import"
    "Platform"="Windows"
    "LicenseType"="AWS"
}

Import-EC2Image -DiskContainer $container @params
```

Saída:

```
Architecture      :
Description       : Windows 2008 Standard Image
Hypervisor        :
ImageId           :
ImportTaskId      : import-ami-abcdefgh
LicenseType       : AWS
Platform          : Windows
Progress          : 2
SnapshotDetails   : {}
Status            : active
StatusMessage     : pending
```

- Para API obter detalhes, consulte [ImportImage](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **ImportKeyPair** com um CLI

Os exemplos de código a seguir mostram como usar o `ImportKeyPair`.

CLI

AWS CLI

Para importar uma chave pública

Primeiro, gere um key pair com a ferramenta de sua escolha. Por exemplo, use este comando `ssh-keygen`:

Comando:

```
ssh-keygen -t rsa -C "my-key" -f ~/.ssh/my-key
```

Saída:

```
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.  
...
```

Este exemplo de comando importa a chave pública especificada.

Comando:

```
aws ec2 import-key-pair --key-name "my-key" --public-key-material fileb://~/.ssh/  
my-key.pub
```

Saída:

```
{
```

```
"KeyName": "my-key",
"KeyFingerprint": "1f:51:ae:28:bf:89:e9:d8:1f:25:5d:37:2d:7d:b8:ca"
}
```

- Para API obter detalhes, consulte [ImportKeyPair](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo importa uma chave pública para EC2. A primeira linha armazena o conteúdo do arquivo de chave pública (*.pub) na variável. **\$publickey** Em seguida, o exemplo converte o UTF8 formato do arquivo de chave pública em uma string codificada em Base64 e armazena a string convertida na variável. **\$pkbase64** Na última linha, a chave pública convertida é importada para EC2. O cmdlet retorna a impressão digital e o nome da chave como resultados.

```
$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
```

Saída:

```
KeyFingerprint                                KeyName
-----
do:d0:15:8f:79:97:12:be:00:fd:df:31:z3:b1:42:z1 Example-user-key
```

- Para API obter detalhes, consulte [ImportKeyPair](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **ImportSnapshot** com um CLI

Os exemplos de código a seguir mostram como usar o `ImportSnapshot`.

CLI

AWS CLI

Para importar um instantâneo

O `import-snapshot` exemplo a seguir importa o disco especificado como um instantâneo.

```
aws ec2 import-snapshot \  
  --description "My server VMDK" \  
  --disk-container Format=VMDK,UserBucket={S3Bucket=my-import-bucket,S3Key=vms/  
my-server-vm.vmdk}
```

Saída:

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

- Para API obter detalhes, consulte [ImportSnapshot](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo importa uma imagem de disco de VM no formato 'VMDK' para um EBS snapshot da Amazon. O exemplo requer uma função de serviço de importação de VM com o nome padrão 'vmimport', com uma política que permita que a EC2 Amazon acesse

o bucket especificado, conforme explicado **VM Import Prerequisites** no tópico em <http://docs.aws.amazon.com/AWSEC2/latest/WindowsGuide/VMImportPrerequisites.html>. Para usar um papel personalizado, especifique o nome do papel usando o **-RoleName** parâmetro.

```
$parms = @{
    "ClientToken"="idempotencyToken"
    "Description"="Disk Image Import"
    "DiskContainer_Description" = "Data disk"
    "DiskContainer_Format" = "VMDK"
    "DiskContainer_S3Bucket" = "amzn-s3-demo-bucket"
    "DiskContainer_S3Key" = "datadiskimage.vmdk"
}

Import-EC2Snapshot @parms
```

Saída:

Description	ImportTaskId	SnapshotTaskDetail
-----	-----	-----
Disk Image Import	import-snap-abcdefgh	Amazon.EC2.Model.SnapshotTaskDetail

- Para API obter detalhes, consulte [ImportSnapshot](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **ModifyCapacityReservation** com um CLI

Os exemplos de código a seguir mostram como usar o `ModifyCapacityReservation`.

CLI

AWS CLI

Exemplo 1: Para alterar o número de instâncias reservadas por uma reserva de capacidade existente

O `modify-capacity-reservation` exemplo a seguir altera o número de instâncias para as quais a reserva de capacidade reserva capacidade.

```
aws ec2 modify-capacity-reservation \  
  --capacity-reservation-id cr-1234abcd56EXAMPLE \  
  --instance-count 5
```

Saída:

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

Exemplo 2: Para alterar a data e a hora de término de uma reserva de capacidade existente

O `modify-capacity-reservation` exemplo a seguir modifica uma reserva de capacidade existente para terminar na data e hora especificadas.

```
aws ec2 modify-capacity-reservation \  
  --capacity-reservation-id cr-1234abcd56EXAMPLE \  
  --end-date-type limited \  
  --end-date 2019-08-31T23:59:59Z
```

Para obter mais informações, consulte [Modificar uma reserva de capacidade](#) no Guia do usuário do Amazon Elastic Compute Cloud para instâncias Linux.

- Para API obter detalhes, consulte [ModifyCapacityReservation](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo modifica o `CapacityReservationId` `cr-0c1f2345db6f7cdba` alterando a contagem de instâncias para 1

```
Edit-EC2CapacityReservation -CapacityReservationId cr-0c1f2345db6f7cdba -  
InstanceCount 1
```

Saída:

True

- Para API obter detalhes, consulte [ModifyCapacityReservation](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **ModifyHosts** com um CLI

Os exemplos de código a seguir mostram como usar o `ModifyHosts`.

CLI

AWS CLI

Exemplo 1: Para habilitar o posicionamento automático para um host dedicado

O `modify-hosts` exemplo a seguir permite o posicionamento automático de um host dedicado para que ele aceite qualquer execução de instância não segmentada que corresponda à configuração do tipo de instância.

```
aws ec2 modify-hosts \  
  --host-id h-06c2f189b4EXAMPLE \  
  --auto-placement on
```

Saída:

```
{  
  "Successful": [  
    "h-06c2f189b4EXAMPLE"  
  ],  
  "Unsuccessful": []  
}
```

Exemplo 2: Para habilitar a recuperação do host para um host dedicado

O `modify-hosts` exemplo a seguir permite a recuperação do host dedicado especificado.

```
aws ec2 modify-hosts \
  --host-id h-06c2f189b4EXAMPLE \
  --host-recovery on
```

Saída:

```
{
  "Successful": [
    "h-06c2f189b4EXAMPLE"
  ],
  "Unsuccessful": []
}
```

Para obter mais informações, consulte [Modificar a colocação automática de hosts dedicados](#) no Guia do usuário do Amazon Elastic Compute Cloud para instâncias Linux.

- Para API obter detalhes, consulte [ModifyHosts](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo modifica as AutoPlacement configurações para desativadas para o host dedicado h-01e23f4cd567890f3

```
Edit-EC2Host -HostId h-03e09f8cd681609f3 -AutoPlacement off
```

Saída:

```
Successful          Unsuccessful
-----
{h-01e23f4cd567890f3} {}
```

- Para API obter detalhes, consulte [ModifyHosts](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **ModifyIdFormat** com um CLI

Os exemplos de código a seguir mostram como usar o `ModifyIdFormat`.

CLI

AWS CLI

Para habilitar o formato de ID mais longo para um recurso

O `modify-id-format` exemplo a seguir ativa o formato de ID mais longo para o tipo de `instance` recurso.

```
aws ec2 modify-id-format \  
  --resource instance \  
  --use-long-ids
```

Para desativar o formato de ID mais longo para um recurso

O `modify-id-format` exemplo a seguir desativa o formato de ID mais longo para o tipo `instance` de recurso.

```
aws ec2 modify-id-format \  
  --resource instance \  
  --no-use-long-ids
```

O `modify-id-format` exemplo a seguir ativa o formato de ID mais longo para todos os tipos de recursos compatíveis que estão dentro do período de aceitação.

```
aws ec2 modify-id-format \  
  --resource all-current \  
  --use-long-ids
```

- Para API obter detalhes, consulte [ModifyIdFormat](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo ativa o formato de ID mais longo para o tipo de recurso especificado.

```
Edit-EC2IdFormat -Resource instance -UseLongId $true
```

Exemplo 2: Este exemplo desativa o formato de ID mais longo para o tipo de recurso especificado.

```
Edit-EC2IdFormat -Resource instance -UseLongId $false
```

- Para API obter detalhes, consulte [ModifyIdFormat](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **ModifyImageAttribute** com um CLI

Os exemplos de código a seguir mostram como usar o `ModifyImageAttribute`.

CLI

AWS CLI

Exemplo 1: Para tornar AMI público

O `modify-instance-attribute` exemplo a seguir torna AMI público o especificado.

```
aws ec2 modify-image-attribute \  
  --image-id ami-5731123e \  
  --launch-permission "Add=[{Group=all}]"
```

Este comando não produz saída.

Exemplo 2: Para tornar um AMI privado

O `modify-instance-attribute` exemplo a seguir torna o especificado AMI privado.

```
aws ec2 modify-image-attribute \  
  --image-id ami-5731123e \  
  --launch-permission "Remove=[{Group=all}]"
```

Este comando não produz saída.

Exemplo 3: para conceder permissão de lançamento a uma AWS conta

O `modify-instance-attribute` exemplo a seguir concede permissões de lançamento à AWS conta especificada.

```
aws ec2 modify-image-attribute \  
  --image-id ami-5731123e \  
  --launch-permission "Add=[{UserId=123456789012}]"
```

Este comando não produz saída.

Exemplo 4: Para remover a permissão de lançamento de uma AWS conta

O `modify-instance-attribute` exemplo a seguir remove as permissões de lançamento da AWS conta especificada.

```
aws ec2 modify-image-attribute \  
  --image-id ami-5731123e \  
  --launch-permission "Remove=[{UserId=123456789012}]"
```

- Para API obter detalhes, consulte [ModifyImageAttribute](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo atualiza a descrição do especificado AMI.

```
Edit-EC2ImageAttribute -ImageId ami-12345678 -Description "New description"
```

Exemplo 2: Este exemplo torna o AMI público (por exemplo, para que qualquer Conta da AWS pessoa possa usá-lo).

```
Edit-EC2ImageAttribute -ImageId ami-12345678 -Attribute launchPermission -  
OperationType add -UserGroup all
```

Exemplo 3: Este exemplo torna o AMI privado (por exemplo, para que somente você, como proprietário, possa usá-lo).


```
Edit-EC2ImageAttribute -ImageId ami-12345678 -Attribute launchPermission -
OperationType remove -UserGroup all
```

Exemplo 4: Este exemplo concede permissão de lançamento ao especificado Conta da AWS.

```
Edit-EC2ImageAttribute -ImageId ami-12345678 -Attribute launchPermission -
OperationType add -UserId 111122223333
```

Exemplo 5: Este exemplo remove a permissão de lançamento do especificado Conta da AWS.

```
Edit-EC2ImageAttribute -ImageId ami-12345678 -Attribute launchPermission -
OperationType remove -UserId 111122223333
```

- Para API obter detalhes, consulte [ModifyImageAttribute](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **ModifyInstanceAttribute** com um CLI

Os exemplos de código a seguir mostram como usar o `ModifyInstanceAttribute`.

CLI

AWS CLI

Exemplo 1: Para modificar o tipo de instância

O `modify-instance-attribute` exemplo a seguir modifica o tipo de instância da instância especificada. A instância deve estar no estado `stopped`.

```
aws ec2 modify-instance-attribute \  
  --instance-id i-1234567890abcdef0 \  
  --instance-type "{\"Value\": \"m1.small\"}"
```

Este comando não produz saída.

Exemplo 2: Para habilitar a rede avançada em uma instância

O `modify-instance-attribute` exemplo a seguir ativa a rede aprimorada para a instância especificada. A instância deve estar no estado `stopped`.

```
aws ec2 modify-instance-attribute \  
  --instance-id i-1234567890abcdef0 \  
  --sriov-net-support simple
```

Este comando não produz saída.

Exemplo 3: Para modificar o `sourceDestCheck` atributo

O `modify-instance-attribute` exemplo a seguir define o `sourceDestCheck` atributo da instância especificada como `true`. A instância deve estar em um VPC.

```
aws ec2 modify-instance-attribute --instance-id i-1234567890abcdef0 --source-  
dest-check "{\"Value\": true}"
```

Este comando não produz saída.

Exemplo 4: Para modificar o `deleteOnTermination` atributo do volume raiz

O `modify-instance-attribute` exemplo a seguir define o `deleteOnTermination` atributo para o volume raiz da instância especificada com EBS suporte da Amazon como `false`. Por padrão, esse atributo é `true` para o volume raiz.

Comando:

```
aws ec2 modify-instance-attribute \  
  --instance-id i-1234567890abcdef0 \  
  --block-device-mappings "[{\"DeviceName\": \"/dev/sda1\", \"Ebs\":  
{\"DeleteOnTermination\": false}}]"
```

Este comando não produz saída.

Exemplo 5: Para modificar os dados do usuário anexados a uma instância

O `modify-instance-attribute` exemplo a seguir adiciona o conteúdo do arquivo `UserData.txt` como `UserData` o da instância especificada.

Conteúdo do arquivo original `UserData.txt`:

```
#!/bin/bash
```

```
yum update -y
service httpd start
chkconfig httpd on
```

O conteúdo do arquivo deve ser codificado em base64. O primeiro comando converte o arquivo de texto em base64 e o salva como um novo arquivo.

Versão Linux/macOS do comando:

```
base64 UserData.txt > UserData.base64.txt
```

Este comando não produz saída.

Versão do comando para Windows:

```
certutil -encode UserData.txt tmp.b64 && findstr /v /c:- tmp.b64 >
UserData.base64.txt
```

Saída:

```
Input Length = 67
Output Length = 152
CertUtil: -encode command completed successfully.
```

Agora você pode referenciar esse arquivo no CLI comando a seguir:

```
aws ec2 modify-instance-attribute \
  --instance-id=i-09b5a14dbca622e76 \
  --attribute userData --value file://UserData.base64.txt
```

Este comando não produz saída.

Para obter mais informações, consulte [Dados do usuário e AWS CLI](#) no Guia EC2 do usuário.

- Para API obter detalhes, consulte [ModifyInstanceAttribute](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Esse exemplo modifica o tipo de instância da instância especificada.

```
Edit-EC2InstanceAttribute -InstanceId i-12345678 -InstanceType m3.medium
```

Exemplo 2: Este exemplo habilita redes aprimoradas para a instância especificada, especificando “simple” como o valor do parâmetro de suporte de rede de virtualização de E/S (SR-IOV) de raiz única, --SriovNetSupport

```
Edit-EC2InstanceAttribute -InstanceId i-12345678 -SriovNetSupport "simple"
```

Exemplo 3: Esse exemplo modifica os grupos de segurança da instância especificada. A instância deve estar em um VPC. Você deve especificar a ID de cada grupo de segurança, não o nome.

```
Edit-EC2InstanceAttribute -InstanceId i-12345678 -Group @( "sg-12345678",  
"sg-45678901" )
```

Exemplo 4: Este exemplo permite a otimização de EBS E/S para a instância especificada. Esse recurso não está disponível em todos os tipos de instância. Cobranças de uso adicionais se aplicam ao usar uma instância EBS otimizada.

```
Edit-EC2InstanceAttribute -InstanceId i-12345678 -EbsOptimized $true
```

Exemplo 5: Este exemplo permite a verificação de origem/destino para a instância especificada. Para que uma NAT instância seja executada NAT, o valor deve ser “false”.

```
Edit-EC2InstanceAttribute -InstanceId i-12345678 -SourceDestCheck $true
```

Exemplo 6: Este exemplo desativa o encerramento da instância especificada.

```
Edit-EC2InstanceAttribute -InstanceId i-12345678 -DisableApiTermination $true
```

Exemplo 7: Esse exemplo altera a instância especificada para que ela seja encerrada quando o desligamento for iniciado a partir da instância.

```
Edit-EC2InstanceAttribute -InstanceId i-12345678 -  
InstanceInitiatedShutdownBehavior terminate
```

- Para API obter detalhes, consulte [ModifyInstanceAttribute](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use `ModifyInstanceCreditSpecification` com um CLI

Os exemplos de código a seguir mostram como usar o `ModifyInstanceCreditSpecification`.

CLI

AWS CLI

Para modificar a opção de crédito para CPU uso de uma instância

Este exemplo modifica a opção de crédito para CPU uso da instância especificada na região especificada para "ilimitada". As opções de crédito válidas são "padrão" e "ilimitadas".

Comando:

```
aws ec2 modify-instance-credit-specification --instance-credit-specification "InstanceId=i-1234567890abcdef0,CpuCredits=unlimited"
```

Saída:

```
{
  "SuccessfulInstanceCreditSpecifications": [
    {
      "InstanceId": "i-1234567890abcdef0"
    }
  ],
  "UnsuccessfulInstanceCreditSpecifications": []
}
```

- Para API obter detalhes, consulte [ModifyInstanceCreditSpecification](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Isso habilita créditos ilimitados de T2, por exemplo, i-01234567890abcdef.

```
$Credit = New-Object -TypeName  
    Amazon.EC2.Model.InstanceCreditSpecificationRequest  
$Credit.InstanceId = "i-01234567890abcdef"  
$Credit.CpuCredits = "unlimited"  
Edit-EC2InstanceCreditSpecification -InstanceCreditSpecification $Credit
```

- Para API obter detalhes, consulte [ModifyInstanceCreditSpecification](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **ModifyNetworkInterfaceAttribute** com um CLI

Os exemplos de código a seguir mostram como usar o `ModifyNetworkInterfaceAttribute`.

CLI

AWS CLI

Para modificar o atributo de anexo de uma interface de rede

Esse exemplo de comando modifica o `attachment` atributo da interface de rede especificada.

Comando:

```
aws ec2 modify-network-interface-attribute --network-interface-id eni-686ea200 --  
attachment AttachmentId=eni-attach-43348162,DeleteOnTermination=false
```

Para modificar o atributo de descrição de uma interface de rede

Esse exemplo de comando modifica o `description` atributo da interface de rede especificada.

Comando:

```
aws ec2 modify-network-interface-attribute --network-interface-id eni-686ea200 --  
description "My description"
```

Para modificar o groupSet atributo de uma interface de rede

Esse exemplo de comando modifica o groupSet atributo da interface de rede especificada.

Comando:

```
aws ec2 modify-network-interface-attribute --network-interface-id eni-686ea200 --  
groups sg-903004f8 sg-1a2b3c4d
```

Para modificar o sourceDestCheck atributo de uma interface de rede

Esse exemplo de comando modifica o sourceDestCheck atributo da interface de rede especificada.

Comando:

```
aws ec2 modify-network-interface-attribute --network-interface-id eni-686ea200 --  
no-source-dest-check
```

- Para API obter detalhes, consulte [ModifyNetworkInterfaceAttribute](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo modifica a interface de rede especificada para que o anexo especificado seja excluído no encerramento.

```
Edit-EC2NetworkInterfaceAttribute -NetworkInterfaceId eni-1a2b3c4d -  
Attachment_AttachmentId eni-attach-1a2b3c4d -Attachment_DeleteOnTermination $true
```

Exemplo 2: Este exemplo modifica a descrição da interface de rede especificada.

```
Edit-EC2NetworkInterfaceAttribute -NetworkInterfaceId eni-1a2b3c4d -Description  
"my description"
```

Exemplo 3: Este exemplo modifica o grupo de segurança da interface de rede especificada.

```
Edit-EC2NetworkInterfaceAttribute -NetworkInterfaceId eni-1a2b3c4d -Groups  
sg-1a2b3c4d
```

Exemplo 4: Este exemplo desativa a verificação de origem/destino para a interface de rede especificada.

```
Edit-EC2NetworkInterfaceAttribute -NetworkInterfaceId eni-1a2b3c4d -  
SourceDestCheck $false
```

- Para API obter detalhes, consulte [ModifyNetworkInterfaceAttribute](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **ModifyReservedInstances** com um CLI

Os exemplos de código a seguir mostram como usar o `ModifyReservedInstances`.

CLI

AWS CLI

Para modificar instâncias reservadas

Esse exemplo de comando move uma instância reservada para outra zona de disponibilidade na mesma região.

Comando:

```
aws ec2 modify-reserved-instances --reserved-instances-ids b847fa93-  
e282-4f55-b59a-1342f5bd7c02 --target-configurations AvailabilityZone=us-  
west-1c,Platform=EC2-Classic,InstanceCount=10
```

Saída:

```
{  
  "ReservedInstancesModificationId": "rimod-d3ed4335-b1d3-4de6-ab31-0f13aaf46687"  
}
```

Para modificar a plataforma de rede das Instâncias Reservadas

Este exemplo de comando converte EC2 -Classic Reserved Instances em -. EC2 VPC

Comando:

```
aws ec2 modify-reserved-instances --reserved-instances-ids f127bd27-  
edb7-44c9-a0eb-0d7e09259af0 --target-configurations AvailabilityZone=us-  
west-1c,Platform=EC2-VPC,InstanceCount=5
```

Saída:

```
{  
  "ReservedInstancesModificationId": "rimod-82fa9020-668f-4fb6-945d-61537009d291"  
}
```

Para obter mais informações, consulte [Modificar suas instâncias reservadas](#) no Guia do EC2 usuário da Amazon.

Para modificar o tamanho da instância das Instâncias Reservadas

Este exemplo de comando modifica uma instância reservada que tem 10 instâncias Linux/m1.small em us-west-1c para que 8 UNIX instâncias m1.small se tornem 2 instâncias m1.large e as 2 m1.small restantes se tornem 1 instância m1.medium na mesma zona de disponibilidade. Comando:

```
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
```

Saída:

```
{  
  "ReservedInstancesModificationId": "rimod-acc5f240-080d-4717-  
b3e3-1c6b11fa00b6"  
}
```

Para obter mais informações, consulte [Modificar o tamanho da instância de suas reservas](#) no Guia do EC2 usuário da Amazon.

- Para API obter detalhes, consulte [ModifyReservedInstances](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Esse exemplo modifica a zona de disponibilidade, a contagem de instâncias e a plataforma das instâncias reservadas especificadas.

```
$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
```

- Para API obter detalhes, consulte [ModifyReservedInstances](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **ModifySnapshotAttribute** com um CLI

Os exemplos de código a seguir mostram como usar o `ModifySnapshotAttribute`.

CLI

AWS CLI

Exemplo 1: Para modificar um atributo de snapshot

O `modify-snapshot-attribute` exemplo a seguir atualiza o `createVolumePermission` atributo do snapshot especificado, removendo as permissões de volume do usuário especificado.

```
aws ec2 modify-snapshot-attribute \
  --snapshot-id snap-1234567890abcdef0 \
  --attribute createVolumePermission \
  --operation-type remove \
```

```
--user-ids 123456789012
```

Exemplo 2: Para tornar público um snapshot

O `modify-snapshot-attribute` exemplo a seguir torna público o snapshot especificado.

```
aws ec2 modify-snapshot-attribute \  
  --snapshot-id snap-1234567890abcdef0 \  
  --attribute createVolumePermission \  
  --operation-type add \  
  --group-names all
```

- Para API obter detalhes, consulte [ModifySnapshotAttribute](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Esse exemplo torna público o snapshot especificado definindo seu `CreateVolumePermission` atributo.

```
Edit-EC2SnapshotAttribute -SnapshotId snap-12345678 -Attribute  
CreateVolumePermission -OperationType Add -GroupName all
```

- Para API obter detalhes, consulte [ModifySnapshotAttribute](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **ModifySpotFleetRequest** com um CLI

Os exemplos de código a seguir mostram como usar o `ModifySpotFleetRequest`.

CLI

AWS CLI

Para modificar uma solicitação de frota spot

Este exemplo de comando atualiza a capacidade alvo da solicitação de frota spot especificada.

Comando:

```
aws ec2 modify-spot-fleet-request --target-capacity 20 --spot-fleet-request-id sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE
```

Saída:

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

Esse exemplo de comando diminui a capacidade alvo da solicitação de frota spot especificada sem, como resultado, encerrar nenhuma instância spot.

Comando:

```
aws ec2 modify-spot-fleet-request --target-capacity 10 --excess-capacity-termination-policy NoTermination --spot-fleet-request-ids sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE
```

Saída:

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

- Para API obter detalhes, consulte [ModifySpotFleetRequest](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo atualiza a capacidade alvo da solicitação de frota spot especificada.

```
Edit-EC2SpotFleetRequest -SpotFleetRequestId sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE -TargetCapacity 10
```

Saída:

```
True
```

- Para API obter detalhes, consulte [ModifySpotFleetRequest](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use `ModifySubnetAttribute` com um CLI

Os exemplos de código a seguir mostram como usar o `ModifySubnetAttribute`.

CLI**AWS CLI**

Para alterar o comportamento de IPv4 endereçamento público de uma sub-rede

Este exemplo modifica a subnet-1a2b3c4d para especificar que todas as instâncias executadas nessa sub-rede recebam um endereço público. IPv4 Se o comando for bem-sucedido, nenhuma saída será retornada.

Comando:

```
aws ec2 modify-subnet-attribute --subnet-id subnet-1a2b3c4d --map-public-ip-on-launch
```

Para alterar o comportamento de IPv6 endereçamento de uma sub-rede

Este exemplo modifica a subnet-1a2b3c4d para especificar que todas as instâncias executadas nessa sub-rede recebam um endereço do intervalo da sub-rede. IPv6

Comando:

```
aws ec2 modify-subnet-attribute --subnet-id subnet-1a2b3c4d --assign-ipv6-address-on-creation
```

Para obter mais informações, consulte o Guia do usuário do endereçamento IP VPC em sua nuvem privada AWS virtual.

- Para API obter detalhes, consulte [ModifySubnetAttribute](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo habilita o endereçamento IP público para a sub-rede especificada.

```
Edit-EC2SubnetAttribute -SubnetId subnet-1a2b3c4d -MapPublicIpOnLaunch $true
```

Exemplo 2: Este exemplo desativa o endereçamento IP público para a sub-rede especificada.

```
Edit-EC2SubnetAttribute -SubnetId subnet-1a2b3c4d -MapPublicIpOnLaunch $false
```

- Para API obter detalhes, consulte [ModifySubnetAttribute](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **ModifyVolumeAttribute** com um CLI

Os exemplos de código a seguir mostram como usar o `ModifyVolumeAttribute`.

CLI

AWS CLI

Para modificar um atributo de volume

Este exemplo define o `autoEnableIo` atributo do volume com o ID `vol-1234567890abcdef0` com `true`. Se o comando for bem-sucedido, nenhuma saída será retornada.

Comando:

```
aws ec2 modify-volume-attribute --volume-id vol-1234567890abcdef0 --auto-enable-io
```

- Para API obter detalhes, consulte [ModifyVolumeAttribute](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo modifica o atributo especificado do volume especificado. As operações de E/S do volume são retomadas automaticamente após serem suspensas devido a dados potencialmente inconsistentes.

```
Edit-EC2VolumeAttribute -VolumeId vol-12345678 -AutoEnableIO $true
```

- Para API obter detalhes, consulte [ModifyVolumeAttribute](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **ModifyVpcAttribute** com um CLI

Os exemplos de código a seguir mostram como usar o `ModifyVpcAttribute`.

CLI

AWS CLI

Para modificar o `enableDnsSupport` atributo

Este exemplo modifica o `enableDnsSupport` atributo. Esse atributo indica se a DNS resolução está habilitada para VPC o. Se esse atributo for `true`, o DNS servidor da Amazon resolverá os DNS nomes de host de suas instâncias para os endereços IP correspondentes; caso contrário, isso não acontece. Se o comando for bem-sucedido, nenhuma saída será retornada.

Comando:

```
aws ec2 modify-vpc-attribute --vpc-id vpc-a01106c2 --enable-dns-support "{\"Value\":false}"
```

Para modificar o enableDnsHostnames atributo

Este exemplo modifica o enableDnsHostnames atributo. Esse atributo indica se as instâncias foram executadas no VPC get DNS hostnames. Se esse atributo for true, instâncias em VPC get DNS hostnames; caso contrário, não o farão. Se o comando for bem-sucedido, nenhuma saída será retornada.

Comando:

```
aws ec2 modify-vpc-attribute --vpc-id vpc-a01106c2 --enable-dns-hostnames "{\"Value\":false}"
```

- Para API obter detalhes, consulte [ModifyVpcAttribute](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo habilita o suporte para DNS nomes de host para o especificadoVPC.

```
Edit-EC2VpcAttribute -VpcId vpc-12345678 -EnableDnsHostnames $true
```

Exemplo 2: Este exemplo desativa o suporte para DNS nomes de host para o especificadoVPC

```
Edit-EC2VpcAttribute -VpcId vpc-12345678 -EnableDnsHostnames $false
```

Exemplo 3: Este exemplo habilita o suporte para DNS resolução para o especificadoVPC.

```
Edit-EC2VpcAttribute -VpcId vpc-12345678 -EnableDnsSupport $true
```

Exemplo 4: Este exemplo desativa o suporte para DNS resolução do especificadoVPC.


```
Edit-EC2VpcAttribute -VpcId vpc-12345678 -EnableDnsSupport $false
```

- Para API obter detalhes, consulte [ModifyVpcAttribute](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **MonitorInstances** com um AWS SDK ou CLI

Os exemplos de código a seguir mostram como usar o `MonitorInstances`.

C++

SDK para C++

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
#!/ Enable detailed monitoring for an Amazon Elastic Compute Cloud (Amazon EC2)
instance.
/*!
 \param instanceId: An EC2 instance ID.
 \param clientConfiguration: AWS client configuration.
 \return bool: Function succeeded.
 */
bool AwsDoc::EC2::enableMonitoring(const Aws::String &instanceId,
                                   const Aws::Client::ClientConfiguration
&clientConfiguration) {
    Aws::EC2::EC2Client ec2Client(clientConfiguration);
    Aws::EC2::Model::MonitorInstancesRequest request;
    request.AddInstanceIds(instanceId);
    request.SetDryRun(true);

    Aws::EC2::Model::MonitorInstancesOutcome dryRunOutcome =
    ec2Client.MonitorInstances(request);
```

```

    if (dryRunOutcome.IsSuccess()) {
        std::cerr
            << "Failed dry run to enable monitoring on instance. A dry run
should trigger an error."
            <<
            std::endl;
        return false;
    } else if (dryRunOutcome.GetError().GetErrorType()
        != Aws::EC2::EC2Errors::DRY_RUN_OPERATION) {
        std::cerr << "Failed dry run to enable monitoring on instance " <<
            instanceId << ": " << dryRunOutcome.GetError().GetMessage() <<
            std::endl;
        return false;
    }

    request.SetDryRun(false);
    Aws::EC2::Model::MonitorInstancesOutcome 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;
    }

    return monitorInstancesOutcome.IsSuccess();
}

```

- Para API obter detalhes, consulte [MonitorInstances](#) em AWS SDK for C++ API Referência.

CLI

AWS CLI

Para habilitar o monitoramento detalhado de uma instância

Este exemplo de comando habilita o monitoramento detalhado da instância especificada.

Comando:

```
aws ec2 monitor-instances --instance-ids i-1234567890abcdef0
```

Saída:

```
{
  "InstanceMonitorings": [
    {
      "InstanceId": "i-1234567890abcdef0",
      "Monitoring": {
        "State": "pending"
      }
    }
  ]
}
```

- Para API obter detalhes, consulte [MonitorInstances](#) na Referência de AWS CLI Comandos.

JavaScript

SDK para JavaScript (v3)

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
import { EC2Client, MonitorInstancesCommand } from "@aws-sdk/client-ec2";

/**
 * 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.
 * @param {{ instanceIds: string[] }} options
 */
export const main = async ({ instanceIds }) => {
  const client = new EC2Client({});
  const command = new MonitorInstancesCommand({
    InstanceIds: instanceIds,
```

```
});

try {
  const { InstanceMonitorings } = await client.send(command);
  const instancesBeingMonitored = InstanceMonitorings.map(
    (im) =>
      ` • Detailed monitoring state for ${im.InstanceId} is
${im.Monitoring.State}.`,
  );
  console.log("Monitoring status:");
  console.log(instancesBeingMonitored.join("\n"));
} catch (caught) {
  if (caught instanceof Error && caught.name === "InvalidParameterValue") {
    console.warn(`${caught.message}`);
  } else {
    throw caught;
  }
}
};
```

- Para API obter detalhes, consulte [MonitorInstances](#) em AWS SDK for JavaScript API Referência.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Esse exemplo permite o monitoramento detalhado da instância especificada.

```
Start-EC2InstanceMonitoring -InstanceId i-12345678
```

Saída:

```
InstanceId      Monitoring
-----
i-12345678     Amazon.EC2.Model.Monitoring
```

- Para API obter detalhes, consulte [MonitorInstances](#) em Referência de AWS Tools for PowerShell cmdlet.

SAP ABAP

SDK para SAP ABAP

 Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
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.
```

- Para API obter detalhes, consulte [MonitorInstances AWS SDK](#) para SAP ABAP API referência.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **MoveAddressToVpc** com um CLI

Os exemplos de código a seguir mostram como usar o `MoveAddressToVpc`.

CLI

AWS CLI

Para mover um endereço para EC2 - VPC

Este exemplo move o endereço IP elástico 54.123.4.56 para a plataforma -. EC2 VPC

Comando:

```
aws ec2 move-address-to-vpc --public-ip 54.123.4.56
```

Saída:

```
{
  "Status": "MoveInProgress"
}
```

- Para API obter detalhes, consulte [MoveAddressToVpc](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: este exemplo move uma EC2 instância com um endereço IP público de 12.345.67.89 para a VPC plataforma EC2 - na região Leste dos EUA (Norte da Virgínia).

```
Move-EC2AddressToVpc -PublicIp 12.345.67.89 -Region us-east-1
```

Exemplo 2: Este exemplo canaliza os resultados de um Get-EC2Instance comando para o Move-EC2AddressToVpc cmdlet. O Get-EC2Instance comando obtém uma instância especificada pelo ID da instância e retorna a propriedade de endereço IP público da instância.

```
(Get-EC2Instance -Instance i-12345678).Instances.PublicIpAddress | Move-EC2AddressToVpc
```

- Para API obter detalhes, consulte [MoveAddressToVpc](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **PurchaseHostReservation** com um CLI

Os exemplos de código a seguir mostram como usar o PurchaseHostReservation.

CLI

AWS CLI

Para comprar uma reserva de anfitrião dedicado

Este exemplo compra a oferta de reserva de host dedicado especificada para o host dedicado especificado em sua conta.

Comando:

```
aws ec2 purchase-host-reservation --offering-id hro-03f707bf363b6b324 --host-id-set h-013abcd2a00cbd123
```

Saída:

```
{
  "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"
}
```

- Para API obter detalhes, consulte [PurchaseHostReservation](#) na Referência de AWS CLI Comandos.

PowerShell**Ferramentas para PowerShell**

Exemplo 1: Este exemplo compra a oferta de reserva hro-0c1f23456789d0ab com configurações que correspondem às do seu host dedicado h-01e23f4cd567890f1

```
New-EC2HostReservation -OfferingId hro-0c1f23456789d0ab HostIdSet
h-01e23f4cd567890f1
```

Saída:

```
ClientToken      :
CurrencyCode     :
Purchase         : {hr-0123f4b5d67bedc89}
TotalHourlyPrice : 1.307
TotalUpfrontPrice : 0.000
```


- Para API obter detalhes, consulte [PurchaseHostReservation](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use `PurchaseScheduledInstances` com um CLI

Os exemplos de código a seguir mostram como usar o `PurchaseScheduledInstances`.

CLI

AWS CLI

Para comprar uma instância programada

Este exemplo compra uma instância programada.

Comando:

```
aws ec2 purchase-scheduled-instances --purchase-requests file://purchase-request.json
```

Purchase-Request.json:

```
[
  {
    "PurchaseToken": "eyJ2IjoiMSIsInMiOiJEsImMiOi...",
    "InstanceCount": 1
  }
]
```

Saída:

```
{
  "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"
  }
]
}

```

- Para API obter detalhes, consulte [PurchaseScheduledInstances](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo compra uma instância programada.

```

$request = New-Object Amazon.EC2.Model.PurchaseRequest
$request.InstanceCount = 1
$request.PurchaseToken = "eyJ2IjoiMSIsInMiOjEsImMiOi..."
New-EC2ScheduledInstancePurchase -PurchaseRequest $request

```

Saída:

```

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
```

- Para API obter detalhes, consulte [PurchaseScheduledInstances](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **RebootInstances** com um AWS SDK ou CLI

Os exemplos de código a seguir mostram como usar o `RebootInstances`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação no contexto no seguinte exemplo de código:

- [Criar e gerenciar um serviço resiliente](#)

.NET

AWS SDK for .NET

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

Reinicialize uma instância pelo ID.

```
/// <summary>
/// Reboot a specific EC2 instance.
/// </summary>
/// <param name="ec2InstanceId">The instance Id of the instance that will be
rebooted.</param>
/// <returns>Async Task.</returns>
public async Task<bool> RebootInstances(string ec2InstanceId)
{
    try
    {
        var request = new RebootInstancesRequest
        {
            InstanceIds = new List<string> { ec2InstanceId },
        };

        await _amazonEC2.RebootInstancesAsync(request);

        // Wait for the instance to be running.
        Console.WriteLine("Waiting for the instance to start.");
        await WaitForInstanceState(ec2InstanceId, InstanceStateName.Running);

        return true;
    }
    catch (AmazonEC2Exception ec2Exception)
    {
        if (ec2Exception.ErrorCode == "InvalidInstanceId")
        {
            _logger.LogError(
                $"InstanceId {ec2InstanceId} is invalid, unable to reboot.
{ec2Exception.Message}");
        }
        return false;
    }
    catch (Exception ex)
    {
        _logger.LogError(
            $"An error occurred while rebooting the instance
{ec2InstanceId}.: {ex.Message}");
        return false;
    }
}

/// <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 in the specified state.
        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.WriteLine(". ");
        } while (!hasState);

        return hasState;
    }

```

Substitua o perfil de uma instância, reinicialize e reinicie um servidor Web.

```

    /// <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)
    {
        try
        {
            await _amazonEc2.ReplaceIamInstanceProfileAssociationAsync(
                new ReplaceIamInstanceProfileAssociationRequest()
                {
                    AssociationId = associationId,
                    IamInstanceProfile = new IamInstanceProfileSpecification()
                    {
                        Name = credsProfileName
                    }
                });
            // Allow time before resetting.
            Thread.Sleep(25000);

            await _amazonEc2.RebootInstancesAsync(
                new RebootInstancesRequest(new List<string>() { instanceId }));
            Thread.Sleep(25000);
            var instanceReady = false;
            var retries = 5;
            while (retries-- > 0 && !instanceReady)
            {
                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("Waiting for instance to be running.");
        }
    }
}
```

```

        await WaitForInstanceState(instanceId, InstanceStateName.Running);
        Console.WriteLine("Instance ready.");
        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}");
    }
    catch (AmazonEC2Exception ec2Exception)
    {
        if (ec2Exception.ErrorCode == "InvalidInstanceID.NotFound")
        {
            _logger.LogError(ec2Exception, $"Instance {instanceId} not
found");
        }

        throw;
    }
    catch (Exception ex)
    {
        _logger.LogError(ex, $"An error occurred while replacing the
template.: {ex.Message}");
        throw;
    }
}

```

- Para API obter detalhes, consulte [RebootInstances](#) em AWS SDK for .NET API Referência.

C++

SDK para C++

 Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
#!/ Reboot an Amazon Elastic Compute Cloud (Amazon EC2) instance.
/*!
 \param instanceID: An EC2 instance ID.
 \param clientConfiguration: AWS client configuration.
 \return bool: Function succeeded.
 */
bool AwsDoc::EC2::rebootInstance(const Aws::String &instanceId,
                                const Aws::Client::ClientConfiguration
&clientConfiguration) {
    Aws::EC2::EC2Client ec2Client(clientConfiguration);

    Aws::EC2::Model::RebootInstancesRequest request;
    request.AddInstanceIds(instanceId);
    request.SetDryRun(true);

    Aws::EC2::Model::RebootInstancesOutcome 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);
```



```
Aws::EC2::Model::RebootInstancesOutcome 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;
}

return outcome.IsSuccess();
}
```

- Para API obter detalhes, consulte [RebootInstances](#) em AWS SDK for C++ API Referência.

CLI

AWS CLI

Para reinicializar uma instância da Amazon EC2

Este exemplo reinicia a instância especificada. Se o comando for bem-sucedido, nenhuma saída será retornada.

Comando:

```
aws ec2 reboot-instances --instance-ids i-1234567890abcdef5
```

Para obter mais informações, consulte Reinicializar a instância no Guia do usuário do Amazon Elastic Compute Cloud.

- Para API obter detalhes, consulte [RebootInstances](#) na Referência de AWS CLI Comandos.

JavaScript

SDK para JavaScript (v3)

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
import { EC2Client, RebootInstancesCommand } from "@aws-sdk/client-ec2";

/**
 * Requests a reboot of the specified instances. This operation is asynchronous;
 * it only queues a request to reboot the specified instances.
 * @param {{ instanceIds: string[] }} options
 */
export const main = async ({ instanceIds }) => {
  const client = new EC2Client({});
  const command = new RebootInstancesCommand({
    InstanceIds: instanceIds,
  });

  try {
    await client.send(command);
    console.log("Instance rebooted successfully.");
  } catch (caught) {
    if (
      caught instanceof Error &&
      caught.name === "InvalidInstanceID.NotFound"
    ) {
      console.warn(
        `${caught.message}. Please provide the InstanceId of a valid instance to
reboot.`
      );
    } else {
      throw caught;
    }
  }
};
```

- Para API obter detalhes, consulte [RebootInstances](#) em AWS SDK for JavaScript API Referência.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo reinicializa a instância especificada.

```
Restart-EC2Instance -InstanceId i-12345678
```

- Para API obter detalhes, consulte [RebootInstances](#) em Referência de AWS Tools for PowerShell cmdlet.

Python

SDK para Python (Boto3)

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
class AutoScalingWrapper:
    """
    Encapsulates Amazon EC2 Auto Scaling and EC2 management actions.
    """

    def __init__(
        self,
        resource_prefix: str,
        inst_type: str,
        ami_param: str,
        autoscaling_client: boto3.client,
        ec2_client: boto3.client,
        ssm_client: boto3.client,
        iam_client: boto3.client,
    ):
        """
```

```
        Initializes the AutoScaler class with the necessary parameters.

        :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
        sts_client = boto3.client("sts")
        self.account_id = sts_client.get_caller_identity()["Account"]

        self.key_pair_name = f"{resource_prefix}-key-pair"
        self.launch_template_name = f"{resource_prefix}-template-"
        self.group_name = f"{resource_prefix}-group"

        # Happy path
        self.instance_policy_name = f"{resource_prefix}-pol"
        self.instance_role_name = f"{resource_prefix}-role"
        self.instance_profile_name = f"{resource_prefix}-prof"

        # Failure mode
        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"

    def replace_instance_profile(
        self,
        instance_id: str,
        new_instance_profile_name: str,
        profile_association_id: str,
    ) -> None:
        """
```

```

    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 restart.
    :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)

        self.ec2_client.reboot_instances(InstanceIds=[instance_id])
        log.info("Rebooting instance %s.", instance_id)
        waiter = self.ec2_client.get_waiter("instance_running")
        log.info("Waiting for instance %s to be running.", instance_id)
        waiter.wait(InstanceIds=[instance_id])
        log.info("Instance %s is now running.", instance_id)

        self.ssm_client.send_command(
            InstanceIds=[instance_id],
            DocumentName="AWS-RunShellScript",
            Parameters={"commands": ["cd / && sudo python3 server.py 80"]},
        )
        log.info(f"Restarted the Python web server on instance
    '{instance_id}'.")
    except ClientError as err:
        log.error("Failed to replace instance profile.")
        error_code = err.response["Error"]["Code"]

```

```

        if error_code == "InvalidAssociationID.NotFound":
            log.error(
                f"Association ID '{profile_association_id}' does not exist."
                "Please check the association ID and try again."
            )
        if error_code == "InvalidInstanceId":
            log.error(
                f"The specified instance ID '{instance_id}' does not exist or
is not available for SSM. "
                f"Please verify the instance ID and try again."
            )
        log.error(f"Full error:\n\t{err}")

```

- Para API obter detalhes, consulte a [RebootInstances](#) Referência AWS SDK do Python (Boto3). API

Rust

SDK para Rust

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```

pub async fn reboot(&self, ec2: &EC2) -> Result<(), EC2Error> {
    if self.instance.is_some() {
        ec2.reboot_instance(self.instance_id()).await?;
        ec2.wait_for_instance_stopped(self.instance_id(), None)
            .await?;
        ec2.wait_for_instance_ready(self.instance_id(), None)
            .await?;
    }
    Ok(())
}

```

```

pub async fn reboot_instance(&self, instance_id: &str) -> Result<(),
EC2Error> {
    tracing::info!("Rebooting instance {instance_id}");

    self.client
        .reboot_instances()
        .instance_ids(instance_id)
        .send()
        .await?;

    Ok(())
}

```

Os garçons, por exemplo, devem estar nos estados parados e prontos, usando os garçons. API Usar os Waiters API requer `use aws_sdk_ec2: :client: :Waiters` no arquivo rust.

```

/// Wait for an instance to be ready and status ok (default wait 60 seconds)
pub async fn wait_for_instance_ready(
    &self,
    instance_id: &str,
    duration: Option<Duration>,
) -> Result<(), EC2Error> {
    self.client
        .wait_until_instance_status_ok()
        .instance_ids(instance_id)
        .wait(duration.unwrap_or(Duration::from_secs(60)))
        .await
        .map_err(|err| match err {
            WaiterError::ExceededMaxWait(exceeded) => EC2Error(format!(
                "Exceeded max time ({}s) waiting for instance to start.",
                exceeded.max_wait().as_secs()
            )),
            _ => EC2Error::from(err),
        })?;
    Ok(())
}

pub async fn wait_for_instance_stopped(
    &self,
    instance_id: &str,
    duration: Option<Duration>,
) -> Result<(), EC2Error> {

```

```

self.client
    .wait_until_instance_stopped()
    .instance_ids(instance_id)
    .wait(duration.unwrap_or(Duration::from_secs(60)))
    .await
    .map_err(|err| match err {
        WaiterError::ExceededMaxWait(exceeded) => EC2Error(format!(
            "Exceeded max time ({}s) waiting for instance to stop.",
            exceeded.max_wait().as_secs(),
        )),
        _ => EC2Error::from(err),
    })?;
Ok(())
}

```

- Para API obter detalhes, consulte [RebootInstances](#) a AWS SDKAPI referência do Rust.

SAP ABAP

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```

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.

```

- Para API obter detalhes, consulte [RebootInstances AWS SDK](#) para SAP ABAP API referência.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **RegisterImage** com um CLI

Os exemplos de código a seguir mostram como usar o RegisterImage.

CLI

AWS CLI

Exemplo 1: Para registrar e AMI usar um arquivo de manifesto

O `register-image` exemplo a seguir registra e AMI usa o arquivo de manifesto especificado no Amazon S3.

```
aws ec2 register-image \  
  --name my-image \  
  --image-location my-s3-bucket/myimage/image.manifest.xml
```

Saída:

```
{  
  "ImageId": "ami-1234567890EXAMPLE"  
}
```

Para obter mais informações, consulte [Amazon Machine Images \(AMI\)](#) no Guia EC2 do usuário da Amazon.

Exemplo 2: Para registrar um AMI usando um instantâneo de um dispositivo raiz

O `register-image` exemplo a seguir registra e AMI usa o snapshot especificado de um volume EBS raiz como dispositivo. `/dev/xvda` O mapeamento do dispositivo de bloco também inclui um EBS volume vazio de 100 GiB como dispositivo. `/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}
```

Saída:

```
{  
  "ImageId": "ami-1a2b3c4d5eEXAMPLE"  
}
```

Para obter mais informações, consulte [Amazon Machine Images \(AMI\)](#) no Guia EC2 do usuário da Amazon.

- Para API obter detalhes, consulte [RegisterImage](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo registra e AMI usa o arquivo de manifesto especificado no Amazon S3.

```
Register-EC2Image -ImageLocation amzn-s3-demo-bucket/my-web-server-ami/  
image.manifest.xml -Name my-web-server-ami
```

- Para API obter detalhes, consulte [RegisterImage](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **RejectVpcPeeringConnection** com um CLI

Os exemplos de código a seguir mostram como usar o `RejectVpcPeeringConnection`.

CLI

AWS CLI

Para rejeitar uma conexão de VPC emparelhamento

Este exemplo rejeita a solicitação de conexão de VPC emparelhamento especificada.

Comando:

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

Saída:

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

- Para API obter detalhes, consulte [RejectVpcPeeringConnection](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: O exemplo acima nega a solicitação de ID de solicitação VpcPeering pcx-01a2b3ce45fe67eb8

```
Deny-EC2VpcPeeringConnection -VpcPeeringConnectionId pcx-01a2b3ce45fe67eb8
```

- Para API obter detalhes, consulte [RejectVpcPeeringConnection](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **ReleaseAddress** com um AWS SDK ou CLI

Os exemplos de código a seguir mostram como usar o ReleaseAddress.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação no contexto no seguinte exemplo de código:

- [Aprender os conceitos básicos](#)

.NET

AWS SDK for .NET

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
/// <summary>  
/// Release an Elastic IP address. After the Elastic IP address is released,  
/// it can no longer be used.  
/// </summary>
```

```
    /// <param name="allocationId">The allocation Id of the Elastic IP address.</  
param>  
    /// <returns>True if successful.</returns>  
    public async Task<bool> ReleaseAddress(string allocationId)  
    {  
        try  
        {  
            var request = new ReleaseAddressRequest { AllocationId =  
allocationId };  
  
            var response = await _amazonEC2.ReleaseAddressAsync(request);  
            return response.HttpStatusCode == HttpStatusCode.OK;  
        }  
        catch (AmazonEC2Exception ec2Exception)  
        {  
            if (ec2Exception.ErrorCode == "InvalidAllocationID.NotFound")  
            {  
                _logger.LogError(  
                    $"AllocationId {allocationId} was not found.  
{ec2Exception.Message}");  
            }  
  
            return false;  
        }  
        catch (Exception ex)  
        {  
            _logger.LogError(  
                $"An error occurred while releasing the AllocationId  
{allocationId}.: {ex.Message}");  
            return false;  
        }  
    }  
}
```

- Para API obter detalhes, consulte [ReleaseAddress](#) em AWS SDK for .NET API Referência.

Bash

AWS CLI com script Bash

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
#####
# 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
}

```

As funções utilitárias usadas neste exemplo.

```

#####
# 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
}
```

- Para API obter detalhes, consulte [ReleaseAddress](#) na Referência de AWS CLI Comandos.

C++

SDK para C++

 Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
#!/ Release an Elastic IP address.
/*!
 \param clientConfiguration: AWS client configuration.
 \return bool: Function succeeded.
 */
bool AwsDoc::EC2::releaseAddress(const Aws::String &allocationID,
                                const Aws::Client::ClientConfiguration
                                &clientConfiguration) {
    Aws::EC2::EC2Client ec2(clientConfiguration);

    Aws::EC2::Model::ReleaseAddressRequest request;
    request.SetAllocationId(allocationID);

    Aws::EC2::Model::ReleaseAddressOutcome 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;
    }

    return outcome.IsSuccess();
}
```

- Para API obter detalhes, consulte [ReleaseAddress](#) em AWS SDK for C++ API Referência.

CLI

AWS CLI

Para liberar um endereço IP elástico para EC2 -Classic

Este exemplo libera um endereço IP elástico para uso com instâncias em EC2 -Classic. Se o comando for bem-sucedido, nenhuma saída será retornada.

Comando:

```
aws ec2 release-address --public-ip 198.51.100.0
```

Para liberar um endereço IP elástico para EC2 - VPC

Este exemplo libera um endereço IP elástico para uso com instâncias em umVPC. Se o comando for bem-sucedido, nenhuma saída será retornada.

Comando:

```
aws ec2 release-address --allocation-id eipalloc-64d5890a
```

- Para API obter detalhes, consulte [ReleaseAddress](#) na Referência de AWS CLI Comandos.

Java

SDK para Java 2.x

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
/**
 * Releases an Elastic IP address asynchronously.
 *
 * @param allocId the allocation ID of the Elastic IP address to be released
 * @return a {@link CompletableFuture} representing the asynchronous
 * operation of releasing the Elastic IP address
 */
```

```
public CompletableFuture<ReleaseAddressResponse>
releaseEC2AddressAsync(String allocId) {
    ReleaseAddressRequest request = ReleaseAddressRequest.builder()
        .allocationId(allocId)
        .build();

    CompletableFuture<ReleaseAddressResponse> response =
getAsyncClient().releaseAddress(request);
    response.whenComplete((resp, ex) -> {
        if (ex != null) {
            throw new RuntimeException("Failed to release Elastic IP
address", ex);
        }
    });

    return response;
}
```

- Para API obter detalhes, consulte [ReleaseAddress](#) em AWS SDK for Java 2.x API Referência.

JavaScript

SDK para JavaScript (v3)

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
import { ReleaseAddressCommand, EC2Client } from "@aws-sdk/client-ec2";

/**
 * Release an Elastic IP address.
 * @param {{ allocationId: string }} options
 */
export const main = async ({ allocationId }) => {
    const client = new EC2Client({});
    const command = new ReleaseAddressCommand({
```

```
// You can also use PublicIp, but that is for EC2 classic which is being
retired.
AllocationId: allocationId,
});

try {
    await client.send(command);
    console.log("Successfully released address.");
} catch (caught) {
    if (
        caught instanceof Error &&
        caught.name === "InvalidAllocationID.NotFound"
    ) {
        console.warn(`${caught.message}. Please provide a valid AllocationID.`);
    } else {
        throw caught;
    }
}
};
```

- Para API obter detalhes, consulte [ReleaseAddress](#) em AWS SDK for JavaScript API Referência.

Kotlin

SDK para Kotlin

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
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")
    }
}
```

- Para API obter detalhes, consulte a [ReleaseAddress](#) referência AWS SDK do Kotlin API.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo libera o endereço IP elástico especificado para instâncias em um VPC.

```
Remove-EC2Address -AllocationId eipalloc-12345678 -Force
```

Exemplo 2: Este exemplo libera o endereço IP elástico especificado para instâncias em EC2 - Classic.

```
Remove-EC2Address -PublicIp 198.51.100.2 -Force
```

- Para API obter detalhes, consulte [ReleaseAddress](#) em Referência de AWS Tools for PowerShell cmdlet.

Python

SDK para Python (Boto3)

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

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

```

class ElasticIp:
    """Represents an Elastic IP and its associated instance."""

    def __init__(
        self, allocation_id: str, public_ip: str, instance_id: Optional[str]
= None
    ) -> None:
        """
        Initializes the ElasticIp object.

        :param allocation_id: The allocation ID of the Elastic IP.
        :param public_ip: The public IP address of the Elastic IP.
        :param instance_id: The ID of the associated EC2 instance, if any.
        """
        self.allocation_id = allocation_id
        self.public_ip = public_ip
        self.instance_id = instance_id

    def __init__(self, ec2_client: Any) -> None:
        """
        Initializes the ElasticIpWrapper with an EC2 client.

        :param ec2_client: A Boto3 Amazon EC2 client. This client provides low-
level
            access to AWS EC2 services.
        """
        self.ec2_client = ec2_client
        self.elastic_ips: List[ElasticIpWrapper.ElasticIp] = []

    @classmethod
    def from_client(cls) -> "ElasticIpWrapper":
        """
        Creates an ElasticIpWrapper instance with a default EC2 client.

        :return: An instance of ElasticIpWrapper initialized with the default EC2
client.
        """
        ec2_client = boto3.client("ec2")
        return cls(ec2_client)

    def release(self, allocation_id: str) -> None:
        """
        Releases an Elastic IP address. After the Elastic IP address is released,

```

```
it can no longer be used.

:param allocation_id: The allocation ID of the Elastic IP to release.
:raises ClientError: If the release fails, such as when the Elastic IP
address is not found.
"""
    elastic_ip = self.get_elastic_ip_by_allocation(self.elastic_ips,
allocation_id)
    if elastic_ip is None:
        logger.info(f"No Elastic IP found with allocation ID
{allocation_id}.")
        return

    try:
        self.ec2_client.release_address(AllocationId=allocation_id)
        self.elastic_ips.remove(elastic_ip) # Remove the Elastic IP from the
list
    except ClientError as err:
        if err.response["Error"]["Code"] == "InvalidAddress.NotFound":
            logger.error(
                f"Failed to release Elastic IP address {allocation_id} "
                "because it could not be found. Verify the Elastic IP address
"
                "and ensure it is allocated to your account in the correct
region "
                "before attempting to release it."
            )
            raise
```

- Para API obter detalhes, consulte a [ReleaseAddress](#) Referência AWS SDK do Python (Boto3). API

Ruby

SDK para Ruby

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
# 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)
  true
rescue StandardError => e
  puts("Error releasing Elastic IP address: #{e.message}")
  false
end
```

- Para API obter detalhes, consulte [ReleaseAddress](#) em AWS SDK for Ruby API Referência.

Rust

SDK para Rust

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
pub async fn deallocate_ip_address(&self, allocation_id: &str) -> Result<(),
EC2Error> {
    self.client
        .release_address()
        .allocation_id(allocation_id)
        .send()
        .await?;
    Ok(())
}
```

- Para API obter detalhes, consulte [ReleaseAddressa AWS SDKAPIreferência do Rust](#).

SAP ABAP

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
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.
```

- Para API obter detalhes, consulte [ReleaseAddress AWS SDK](#) para SAP ABAP API referência.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **ReleaseHosts** com um CLI

Os exemplos de código a seguir mostram como usar o `ReleaseHosts`.

CLI

AWS CLI

Para liberar um host dedicado da sua conta

Para liberar um host dedicado da sua conta. As instâncias que estão no host devem ser interrompidas ou encerradas antes que o host possa ser liberado.

Comando:

```
aws ec2 release-hosts --host-id=h-0029d6e3cacf1b3da
```

Saída:

```
{
  "Successful": [
    "h-0029d6e3cacf1b3da"
  ],
  "Unsuccessful": []
}
```

- Para API obter detalhes, consulte [ReleaseHosts](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo libera o ID de host fornecido h-0badafd1dcb2f3456

```
Remove-EC2Host -HostId h-0badafd1dcb2f3456
```

Saída:

```
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} {}
```

- Para API obter detalhes, consulte [ReleaseHosts](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **ReplaceIamInstanceProfileAssociation** com um AWS SDK ou CLI

Os exemplos de código a seguir mostram como usar o `ReplaceIamInstanceProfileAssociation`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação no contexto no seguinte exemplo de código:

- [Criar e gerenciar um serviço resiliente](#)

.NET

AWS SDK for .NET

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
/// <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)
{
    try
    {
        await _amazonEc2.ReplaceIamInstanceProfileAssociationAsync(
            new ReplaceIamInstanceProfileAssociationRequest()
            {
                AssociationId = associationId,
                IamInstanceProfile = new IamInstanceProfileSpecification()
                {
                    Name = credsProfileName
                }
            });
        // Allow time before resetting.
        Thread.Sleep(25000);

        await _amazonEc2.RebootInstancesAsync(
            new RebootInstancesRequest(new List<string>() { instanceId }));
    }
}
```

```

Thread.Sleep(25000);
var instanceReady = false;
var retries = 5;
while (retries-- > 0 && !instanceReady)
{
    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("Waiting for instance to be running.");
await WaitForInstanceState(instanceId, InstanceStateName.Running);
Console.WriteLine("Instance ready.");
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}");
}
catch (AmazonEC2Exception ec2Exception)
{
    if (ec2Exception.ErrorCode == "InvalidInstanceID.NotFound")

```

```

        {
            _logger.LogError(ec2Exception, $"Instance {instanceId} not
found");
        }

        throw;
    }
    catch (Exception ex)
    {
        _logger.LogError(ex, $"An error occurred while replacing the
template.: {ex.Message}");
        throw;
    }
}

```

- Para API obter detalhes, consulte [ReplacelamInstanceProfileAssociation](#) em AWS SDK for .NET API Referência.

CLI

AWS CLI

Para substituir um perfil de IAM instância por uma instância

Este exemplo substitui o perfil da IAM instância representado pela associação `iip-
assoc-060bae234aac2e7fa` pelo perfil da IAM instância chamado `AdminRole`.

```

aws ec2 replace-iam-instance-profile-association \
  --iam-instance-profile Name=AdminRole \
  --association-id iip-assoc-060bae234aac2e7fa

```

Saída:

```

{
  "IamInstanceProfileAssociation": {
    "InstanceId": "i-087711ddaf98f9489",
    "State": "associating",
    "AssociationId": "iip-assoc-0b215292fab192820",
    "IamInstanceProfile": {
      "Id": "AIPAJLNLDX3AMYZNWYYAY",
      "Arn": "arn:aws:iam::123456789012:instance-profile/AdminRole"
    }
  }
}

```

```
    }  
  }  
}
```

- Para API obter detalhes, consulte [ReplacelamInstanceProfileAssociation](#) na Referência de AWS CLI Comandos.

JavaScript

SDK para JavaScript (v3)

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
await retry({ intervalInMs: 1000, maxRetries: 30 }, () =>  
  ec2Client.send(  
    new ReplaceIamInstanceProfileAssociationCommand({  
      AssociationId: state.instanceProfileAssociationId,  
      IamInstanceProfile: { Name: NAMES.ssmOnlyInstanceProfileName },  
    })),  
  ),  
);
```

- Para API obter detalhes, consulte [ReplacelamInstanceProfileAssociation](#) em AWS SDK for JavaScript API Referência.

Python

SDK para Python (Boto3)

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

Este exemplo substitui o perfil de instância de uma instância em execução, reinicia a instância e envia um comando para a instância após ela iniciar.

```
class AutoScalingWrapper:
    """
    Encapsulates Amazon EC2 Auto Scaling and EC2 management actions.
    """

    def __init__(
        self,
        resource_prefix: str,
        inst_type: str,
        ami_param: str,
        autoscaling_client: boto3.client,
        ec2_client: boto3.client,
        ssm_client: boto3.client,
        iam_client: boto3.client,
    ):
        """
        Initializes the AutoScaler class with the necessary parameters.

        :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
        sts_client = boto3.client("sts")
        self.account_id = sts_client.get_caller_identity()["Account"]

        self.key_pair_name = f"{resource_prefix}-key-pair"
        self.launch_template_name = f"{resource_prefix}-template-"
        self.group_name = f"{resource_prefix}-group"
```



```
# Happy path
self.instance_policy_name = f"{resource_prefix}-pol"
self.instance_role_name = f"{resource_prefix}-role"
self.instance_profile_name = f"{resource_prefix}-prof"

# Failure mode
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"

def replace_instance_profile(
    self,
    instance_id: str,
    new_instance_profile_name: str,
    profile_association_id: str,
) -> None:
    """
    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 restart.
    :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)

self.ec2_client.reboot_instances(InstanceIds=[instance_id])
log.info("Rebooting instance %s.", instance_id)
waiter = self.ec2_client.get_waiter("instance_running")
log.info("Waiting for instance %s to be running.", instance_id)
waiter.wait(InstanceIds=[instance_id])
log.info("Instance %s is now running.", instance_id)

self.ssm_client.send_command(
    InstanceIds=[instance_id],
    DocumentName="AWS-RunShellScript",
    Parameters={"commands": ["cd / && sudo python3 server.py 80"]},
)
log.info(f"Restarted the Python web server on instance
'{instance_id}'.")
except ClientError as err:
    log.error("Failed to replace instance profile.")
    error_code = err.response["Error"]["Code"]
    if error_code == "InvalidAssociationID.NotFound":
        log.error(
            f"Association ID '{profile_association_id}' does not exist."
            "Please check the association ID and try again."
        )
    if error_code == "InvalidInstanceId":
        log.error(
            f"The specified instance ID '{instance_id}' does not exist or
is not available for SSM. "
            f"Please verify the instance ID and try again."
        )
    log.error(f"Full error:\n\t{err}")
```

- Para API obter detalhes, consulte a [ReplacelamInstanceProfileAssociation](#) Referência AWS SDK do Python (Boto3). API

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use `ReplaceNetworkAclAssociation` com um CLI

Os exemplos de código a seguir mostram como usar o `ReplaceNetworkAclAssociation`.

CLI

AWS CLI

Para substituir a rede ACL associada a uma sub-rede

Este exemplo associa a rede especificada ACL à sub-rede da associação de rede ACL especificada.

Comando:

```
aws ec2 replace-network-acl-association --association-id aclassoc-e5b95c8c --  
network-acl-id acl-5fb85d36
```

Saída:

```
{  
  "NewAssociationId": "aclassoc-3999875b"  
}
```

- Para API obter detalhes, consulte [ReplaceNetworkAclAssociation](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo associa a rede especificada à ACL sub-rede da associação de rede ACL especificada.

```
Set-EC2NetworkAclAssociation -NetworkAclId acl-12345678 -AssociationId  
aclassoc-1a2b3c4d
```

Saída:

```
aclassoc-87654321
```

- Para API obter detalhes, consulte [ReplaceNetworkAclAssociation](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use `ReplaceNetworkAclEntry` com um CLI

Os exemplos de código a seguir mostram como usar o `ReplaceNetworkAclEntry`.

CLI

AWS CLI

Para substituir uma ACL entrada de rede

Este exemplo substitui uma entrada para a rede ACL especificada. A nova regra 100 permite o tráfego de entrada de 203.0.113.12/24 na UDP porta 53 () em qualquer sub-rede associada.
DNS

Comando:

```
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
```

- Para API obter detalhes, consulte [ReplaceNetworkAclEntry](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo substitui a entrada especificada para a rede ACL especificada. A nova regra permite tráfego de entrada do endereço especificado para qualquer sub-rede associada.

```
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
```

- Para API obter detalhes, consulte [ReplaceNetworkAclEntry](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **ReplaceRoute** com um CLI

Os exemplos de código a seguir mostram como usar o `ReplaceRoute`.

CLI

AWS CLI

Para substituir uma rota

Este exemplo substitui a rota especificada na tabela de rotas especificada. A nova rota corresponde à especificada CIDR e envia o tráfego para o gateway privado virtual especificado. Se o comando for bem-sucedido, nenhuma saída será retornada.

Comando:

```
aws ec2 replace-route --route-table-id rtb-22574640 --destination-cidr-  
block 10.0.0.0/16 --gateway-id vgw-9a4cacf3
```

- Para API obter detalhes, consulte [ReplaceRoute](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo substitui a rota especificada pela tabela de rotas especificada. A nova rota envia o tráfego especificado para o gateway privado virtual especificado.

```
Set-EC2Route -RouteTableId rtb-1a2b3c4d -DestinationCidrBlock 10.0.0.0/24 -  
GatewayId vgw-1a2b3c4d
```

- Para API obter detalhes, consulte [ReplaceRoute](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use `ReplaceRouteTableAssociation` com um CLI

Os exemplos de código a seguir mostram como usar o `ReplaceRouteTableAssociation`.

CLI

AWS CLI

Para substituir a tabela de rotas associada a uma sub-rede

Este exemplo associa a tabela de rotas especificada à sub-rede para a associação da tabela de rotas especificada.

Comando:

```
aws ec2 replace-route-table-association --association-id rtbassoc-781d0d1a --  
route-table-id rtb-22574640
```

Saída:

```
{  
  "NewAssociationId": "rtbassoc-3a1f0f58"  
}
```

- Para API obter detalhes, consulte [ReplaceRouteTableAssociation](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo associa a tabela de rotas especificada à sub-rede para a associação da tabela de rotas especificada.

```
Set-EC2RouteTableAssociation -RouteTableId rtb-1a2b3c4d -AssociationId  
rtbassoc-12345678
```

Saída:

```
rtbassoc-87654321
```

- Para API obter detalhes, consulte [ReplaceRouteTableAssociation](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **ReportInstanceStatus** com um CLI

Os exemplos de código a seguir mostram como usar o ReportInstanceStatus.

CLI

AWS CLI

Para relatar feedback de status de uma instância

Este exemplo de comando relata o feedback de status da instância especificada.

Comando:

```
aws ec2 report-instance-status --instances i-1234567890abcdef0 --status impaired  
--reason-codes unresponsive
```

- Para API obter detalhes, consulte [ReportInstanceStatus](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo relata o feedback de status da instância especificada.

```
Send-EC2InstanceStatus -Instance i-12345678 -Status impaired -ReasonCode unresponsive
```

- Para API obter detalhes, consulte [ReportInstanceStatus](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **RequestSpotFleet** com um CLI

Os exemplos de código a seguir mostram como usar o RequestSpotFleet.

CLI

AWS CLI

Para solicitar uma frota spot na sub-rede com o menor preço

Esse exemplo de comando cria uma solicitação de frota Spot com duas especificações de lançamento que diferem somente por sub-rede. A frota Spot executa as instâncias na sub-rede especificada com o menor preço. Se as instâncias forem iniciadas em um padrãoVPC, elas receberão um endereço IP público por padrão. Se as instâncias forem executadas de forma não padrãoVPC, elas não receberão um endereço IP público por padrão.

Observe que você não pode especificar sub-redes diferentes da mesma zona de disponibilidade em uma solicitação de frota spot.

Comando:

```
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"
      }
    }
  ]
}
```

Saída:

```
{
  "SpotFleetRequestId": "sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE"
}
```

Para solicitar uma frota Spot na Zona de Disponibilidade com o menor preço

Esse exemplo de comando cria uma solicitação de frota spot com duas especificações de lançamento que diferem somente por zona de disponibilidade. A frota spot lança as instâncias na zona de disponibilidade especificada com o menor preço. Se sua conta suportar VPC apenas EC2 -, a Amazon EC2 executa as instâncias spot na sub-rede padrão da zona de disponibilidade. Se sua conta suportar EC2 -Classic, a Amazon EC2 iniciará as instâncias em EC2 -Classic na zona de disponibilidade.

Comando:

```
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"
      }
    }
  ]
}
```

Para iniciar instâncias spot em uma sub-rede e atribuir a elas endereços IP públicos

Este exemplo de comando atribui endereços públicos às instâncias executadas em um formato não VPC padrão. Observe que, ao especificar uma interface de rede, você deve incluir a ID da sub-rede e a ID do grupo de segurança usando a interface de rede.

Comando:

```
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"
    }
  }
]
}

```

Para solicitar uma frota Spot usando a estratégia de alocação diversificada

Esse exemplo de comando cria uma solicitação de frota spot que executa 30 instâncias usando a estratégia de alocação diversificada. As especificações de lançamento variam de acordo com o tipo de instância. A frota Spot distribui as instâncias de acordo com as especificações de lançamento, de forma que haja 10 instâncias de cada tipo.

Comando:

```
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"
    }
]
}

```

Para obter mais informações, consulte Solicitações de frota spot no Guia do usuário do Amazon Elastic Compute Cloud.

- Para API obter detalhes, consulte [RequestSpotFleet](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Esse exemplo cria uma solicitação de frota spot na zona de disponibilidade com o menor preço para o tipo de instância especificado. Se sua conta suportar VPC apenas EC2 -, a frota spot executa as instâncias na zona de disponibilidade de menor preço que tem uma sub-rede padrão. Se sua conta suportar EC2 -Classic, a frota spot executa as instâncias em EC2 -Classic na zona de disponibilidade de menor preço. Observe que o preço pago não excederá o preço spot especificado para a solicitação.

```

$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

```

- Para API obter detalhes, consulte [RequestSpotFleet](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **RequestSpotInstances** com um CLI

Os exemplos de código a seguir mostram como usar o RequestSpotInstances.

CLI

AWS CLI

Para solicitar instâncias spot

Este exemplo de comando cria uma solicitação única de instância spot para cinco instâncias na zona de disponibilidade especificada. Se sua conta suportar VPC apenas EC2 -, a Amazon EC2 executa as instâncias na sub-rede padrão da zona de disponibilidade especificada. Se sua conta suportar EC2 -Classic, a Amazon EC2 executará as instâncias em EC2 -Classic na zona de disponibilidade especificada.

Comando:

```
aws ec2 request-spot-instances --spot-price "0.03" --instance-count 5 --  
type "one-time" --launch-specification file://specification.json
```

Especificação.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"  
  }  
}
```

```
}
```

Saída:

```
{
  "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"
    },
    ...
  ]
}
```

Esse exemplo de comando cria uma solicitação única de instância spot para cinco instâncias na sub-rede especificada. A Amazon EC2 lança as instâncias na sub-rede especificada. Se não VPC for padrãoVPC, as instâncias não receberão um endereço IP público por padrão.

Comando:

```
aws ec2 request-spot-instances --spot-price "0.050" --instance-count 5 --  
type "one-time" --launch-specification file://specification.json
```

Especificação.json:

```
{  
  "ImageId": "ami-1a2b3c4d",  
  "SecurityGroupIds": [ "sg-1a2b3c4d" ],  
  "InstanceType": "m3.medium",  
  "SubnetId": "subnet-1a2b3c4d",  
  "IamInstanceProfile": {  
    "Arn": "arn:aws:iam::123456789012:instance-profile/my-iam-role"  
  }  
}
```

Saída:

```
{  
  "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"
},
...
]
}

```

Este exemplo atribui um endereço IP público às Instâncias Spot que você executa de forma não VPC padrão. Observe que, ao especificar uma interface de rede, você deve incluir a ID da sub-rede e a ID do grupo de segurança usando a interface de rede.

Comando:

```
aws ec2 request-spot-instances --spot-price "0.050" --instance-count 1 --
type "one-time" --launch-specification file://specification.json
```

Especificação.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"
    }
}

```

- Para API obter detalhes, consulte [RequestSpotInstances](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo solicita uma instância spot única na sub-rede especificada. Observe que o grupo de segurança deve ser criado para o VPC que contém a sub-rede especificada e deve ser especificado por ID usando a interface de rede. Ao especificar uma interface de rede, você deve incluir a ID da sub-rede usando a interface de rede.

```

$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

```

Saída:

```

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
```

- Para API obter detalhes, consulte [RequestSpotInstances](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **ResetImageAttribute** com um CLI

Os exemplos de código a seguir mostram como usar o `ResetImageAttribute`.

CLI

AWS CLI

Para redefinir o `launchPermission` atributo

Este exemplo redefine o `launchPermission` atributo do especificado AMI para seu valor padrão. Por padrão, AMIs são privados. Se o comando for bem-sucedido, nenhuma saída será retornada.

Comando:

```
aws ec2 reset-image-attribute --image-id ami-5731123e --
attribute launchPermission
```

- Para API obter detalhes, consulte [ResetImageAttribute](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo redefine o atributo `launchPermission` " para seu valor padrão. Por padrão, AMIs são privados.

```
Reset-EC2ImageAttribute -ImageId ami-12345678 -Attribute launchPermission
```

- Para API obter detalhes, consulte [ResetImageAttribute](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **ResetInstanceAttribute** com um CLI

Os exemplos de código a seguir mostram como usar o `ResetInstanceAttribute`.

CLI

AWS CLI

Para redefinir o `sourceDestCheck` atributo

Este exemplo redefine o `sourceDestCheck` atributo da instância especificada. A instância deve estar em um VPC. Se o comando for bem-sucedido, nenhuma saída será retornada.

Comando:

```
aws ec2 reset-instance-attribute --instance-id i-1234567890abcdef0 --  
attribute sourceDestCheck
```

Para redefinir o atributo do kernel

Este exemplo redefine o `kernel` atributo da instância especificada. A instância deve estar no estado `stopped`. Se o comando for bem-sucedido, nenhuma saída será retornada.

Comando:

```
aws ec2 reset-instance-attribute --instance-id i-1234567890abcdef0 --  
attribute kernel
```

Para redefinir o atributo `ramdisk`

Este exemplo redefine o `ramdisk` atributo da instância especificada. A instância deve estar no estado `stopped`. Se o comando for bem-sucedido, nenhuma saída será retornada.

Comando:

```
aws ec2 reset-instance-attribute --instance-id i-1234567890abcdef0 --  
attribute ramdisk
```

- Para API obter detalhes, consulte [ResetInstanceAttribute](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Esse exemplo redefine o atributo sriovNetSupport " para a instância especificada.

```
Reset-EC2InstanceAttribute -InstanceId i-12345678 -Attribute sriovNetSupport
```

Exemplo 2: Esse exemplo redefine o atributo ebsOptimized " para a instância especificada.

```
Reset-EC2InstanceAttribute -InstanceId i-12345678 -Attribute ebsOptimized
```

Exemplo 3: Esse exemplo redefine o atributo sourceDestCheck " para a instância especificada.

```
Reset-EC2InstanceAttribute -InstanceId i-12345678 -Attribute sourceDestCheck
```

Exemplo 4: Esse exemplo redefine o atributo disableApiTermination " para a instância especificada.

```
Reset-EC2InstanceAttribute -InstanceId i-12345678 -Attribute  
disableApiTermination
```

Exemplo 5: Esse exemplo redefine o atributo "instanceInitiatedShutdownComportamento" para a instância especificada.

```
Reset-EC2InstanceAttribute -InstanceId i-12345678 -Attribute  
instanceInitiatedShutdownBehavior
```

- Para API obter detalhes, consulte [ResetInstanceAttribute](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use `ResetNetworkInterfaceAttribute` com um CLI

Os exemplos de código a seguir mostram como usar o `ResetNetworkInterfaceAttribute`.

CLI

AWS CLI

Para redefinir um atributo de interface de rede

O `reset-network-interface-attribute` exemplo a seguir redefine o valor do atributo de verificação de origem/destino para `true`

```
aws ec2 reset-network-interface-attribute \  
  --network-interface-id eni-686ea200 \  
  --source-dest-check
```

Este comando não produz saída.

- Para API obter detalhes, consulte [ResetNetworkInterfaceAttribute](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo redefine a verificação de origem/destino para a interface de rede especificada.

```
Reset-EC2NetworkInterfaceAttribute -NetworkInterfaceId eni-1a2b3c4d -  
SourceDestCheck
```

- Para API obter detalhes, consulte [ResetNetworkInterfaceAttribute](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use `ResetSnapshotAttribute` com um CLI

Os exemplos de código a seguir mostram como usar o `ResetSnapshotAttribute`.

CLI

AWS CLI

Para redefinir um atributo de snapshot

Este exemplo redefine as permissões de criação de volume para `snapshotsnap-1234567890abcdef0`. Se o comando for bem-sucedido, nenhuma saída será retornada.

Comando:

```
aws ec2 reset-snapshot-attribute --snapshot-id snap-1234567890abcdef0 --  
attribute createVolumePermission
```

- Para API obter detalhes, consulte [ResetSnapshotAttribute](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo redefine o atributo especificado do instantâneo especificado.

```
Reset-EC2SnapshotAttribute -SnapshotId snap-12345678 -Attribute  
CreateVolumePermission
```

- Para API obter detalhes, consulte [ResetSnapshotAttribute](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use `RevokeSecurityGroupEgress` com um CLI

Os exemplos de código a seguir mostram como usar o `RevokeSecurityGroupEgress`.

CLI

AWS CLI

Exemplo 1: Para remover a regra que permite tráfego de saída para um intervalo de endereços específico

O comando de `revoke-security-group-egress` exemplo a seguir remove a regra que concede acesso aos intervalos de endereços especificados na TCP porta 80.

```
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"}]}
```

Este comando não produz saída.

Para obter mais informações, consulte [Grupos de segurança](#) no Guia EC2 do usuário da Amazon.

Exemplo 2: Para remover a regra que permite tráfego de saída para um grupo de segurança específico

O comando de `revoke-security-group-egress` exemplo a seguir remove a regra que concede acesso ao grupo de segurança especificado na TCP porta 80.

```
aws ec2 revoke-security-group-egress \  
  --group-id sg-026c12253ce15eff7 \  
  --ip-permissions '[{"IpProtocol": "tcp", "FromPort": 443, "ToPort": 443, "UserIdGroupPairs": [{"GroupId": "sg-06df23a01ff2df86d"}]}]'
```

Este comando não produz saída.

Para obter mais informações, consulte [Grupos de segurança](#) no Guia EC2 do usuário da Amazon.

- Para API obter detalhes, consulte [RevokeSecurityGroupEgress](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo remove a regra do grupo de segurança especificado para EC2 - VPC. Isso revoga o acesso ao intervalo de endereços IP especificado na TCP porta 80. A sintaxe usada neste exemplo requer a PowerShell versão 3 ou superior.

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

Exemplo 2: Com a PowerShell versão 2, você deve usar `New-Object` para criar o `IpPermission` objeto.

```
$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
```

Exemplo 3: Este exemplo revoga o acesso ao grupo de segurança de origem especificado na TCP porta 80.

```
$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 } )
```

- Para API obter detalhes, consulte [RevokeSecurityGroupEgress](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use `RevokeSecurityGroupIngress` com um CLI

Os exemplos de código a seguir mostram como usar o `RevokeSecurityGroupIngress`.

CLI

AWS CLI

Exemplo 1: Para remover uma regra de um grupo de segurança

O `revoke-security-group-ingress` exemplo a seguir remove o acesso à TCP porta 22 para o intervalo de `203.0.113.0/24` endereços do grupo de segurança especificado como padrãoVPC.

```
aws ec2 revoke-security-group-ingress \
  --group-name mySecurityGroup \
  --protocol tcp \
  --port 22 \
  --cidr 203.0.113.0/24
```

Esse comando não produzirá nenhuma saída se for bem-sucedido.

Para obter mais informações, consulte [Grupos de segurança](#) no Guia EC2 do usuário da Amazon.

Exemplo 2: Para remover uma regra usando o conjunto de permissões de IP

O `revoke-security-group-ingress` exemplo a seguir usa o `ip-permissions` parâmetro para remover uma regra de entrada que permite a ICMP mensagem Destination Unreachable: Fragmentation Needed and Don't Fragment was Set (Tipo 3, Código 4).

```
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}]
```

Esse comando não produzirá nenhuma saída se for bem-sucedido.

Para obter mais informações, consulte [Grupos de segurança](#) no Guia EC2 do usuário da Amazon.

- Para API obter detalhes, consulte [RevokeSecurityGroupIngress](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo revoga o acesso à TCP porta 22 do intervalo de endereços especificado para o grupo de segurança especificado para EC2 -VPC. Observe que você deve identificar grupos de segurança para EC2 VPC usar o ID do grupo de segurança e não o nome do grupo de segurança. A sintaxe usada neste exemplo requer a PowerShell versão 3 ou superior.

```
$ip = @{ IpProtocol="tcp"; FromPort="22"; ToPort="22";  
  IpRanges="203.0.113.0/24" }  
Revoke-EC2SecurityGroupIngress -GroupId sg-12345678 -IpPermission $ip
```

Exemplo 2: Com a PowerShell versão 2, você deve usar New-Object para criar o IpPermission objeto.

```
$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
```

Exemplo 3: Este exemplo revoga o acesso à TCP porta 22 do intervalo de endereços especificado para o grupo de segurança especificado para EC2 -Classic. A sintaxe usada neste exemplo requer a PowerShell versão 3 ou superior.

```
$ip = @{ IpProtocol="tcp"; FromPort="22"; ToPort="22";  
  IpRanges="203.0.113.0/24" }  
  
Revoke-EC2SecurityGroupIngress -GroupName "my-security-group" -IpPermission $ip
```

Exemplo 4: Com a PowerShell versão 2, você deve usar `New-Object` para criar o `IpPermission` objeto.

```
$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
```

- Para API obter detalhes, consulte [RevokeSecurityGroupIngress](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use `RunInstances` com um AWS SDK ou CLI

Os exemplos de código a seguir mostram como usar o `RunInstances`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação no contexto no seguinte exemplo de código:

- [Aprender os conceitos básicos](#)

.NET

AWS SDK for .NET

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
/// <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)
    {
        try
        {
            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);
            var instanceId = response.Reservation.Instances[0].InstanceId;

            Console.WriteLine("Waiting for the instance to start.");
            await WaitForInstanceState(instanceId, InstanceStateName.Running);

            return instanceId;
        }
        catch (AmazonEC2Exception ec2Exception)
        {
            if (ec2Exception.ErrorCode == "InvalidGroupId.NotFound")
            {
                _logger.LogError(
                    $"GroupId {groupId} was not found. {ec2Exception.Message}");
            }

            throw;
        }
        catch (Exception ex)
        {
```

```

        _logger.LogError(
            $"An error occurred while running the instance.: {ex.Message}");
        throw;
    }
}

```

- Para API obter detalhes, consulte [RunInstances](#) em AWS SDK for .NET API Referência.

Bash

AWS CLI com script Bash

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```

#####
# 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 getopt 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
}

```

As funções utilitárias usadas neste exemplo.

```

#####
# 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
}

```

- Para API obter detalhes, consulte [RunInstances](#) na Referência de AWS CLI Comandos.

C++

SDK para C++

 Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```

//! Launch an Amazon Elastic Compute Cloud (Amazon EC2) instance.
/*!
 \param instanceName: A name for the EC2 instance.
 \param amiId: An Amazon Machine Image (AMI) identifier.
 \param[out] instanceID: String to return the instance ID.
 \param clientConfiguration: AWS client configuration.
 \return bool: Function succeeded.
 */
bool AwsDoc::EC2::runInstance(const Aws::String &instanceName,
                             const Aws::String &amiId,
                             Aws::String &instanceID,
                             const Aws::Client::ClientConfiguration
&clientConfiguration) {
    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();

    return true;
}

```

- Para API obter detalhes, consulte [RunInstances](#) em AWS SDK for C++ API Referência.

CLI

AWS CLI

Exemplo 1: executar uma instância em uma sub-rede padrão

O `run-instances` exemplo a seguir inicia uma única instância do tipo `t2.micro` na sub-rede padrão da região atual e a associa à sub-rede padrão VPC da região. O key pair é opcional se você não planeja se conectar à sua instância usando SSH (Linux) ou RDP (Windows).

```

aws ec2 run-instances \
  --image-id ami-0abcdef1234567890 \
  --instance-type t2.micro \
  --key-name MyKeyPair

```

Saída:

```

{
  "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"  
}
```

Exemplo 2: para executar uma instância em uma sub-rede não padrão e adicionar um endereço IP público

O exemplo `run-instances` a seguir solicita um endereço IP público para uma instância que você está executando em uma sub-rede não padrão. A instância está associada ao grupo de segurança especificado.

```
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
```

Para obter um exemplo da saída de `run-instances`, consulte o Exemplo 1.

Exemplo 3: executar uma instância com volumes adicionais

O exemplo `run-instances` a seguir usa um mapeamento de dispositivos de blocos, especificado em `mapping.json`, para anexar volumes adicionais na execução. Um mapeamento de dispositivos de blocos pode especificar EBS volumes, volumes de armazenamento de instâncias ou EBS volumes e volumes de armazenamento de instâncias.

```
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
```

Conteúdo de `mapping.json`. Este exemplo adiciona `/dev/sdh` um EBS volume vazio com um tamanho de 100 GiB.

```
[
  {
    "DeviceName": "/dev/sdh",
    "Ebs": {
      "VolumeSize": 100
    }
  }
]
```

Conteúdo de `mapping.json`. Este exemplo adiciona `ephemeral1` como um volume de armazenamento de instância.

```
[
  {
    "DeviceName": "/dev/sdc",
    "VirtualName": "ephemeral1"
  }
]
```

Para obter um exemplo da saída de `run-instances`, consulte o Exemplo 1.

Para obter mais informações sobre mapeamentos de dispositivos de blocos, consulte [Mapeamento de dispositivos de blocos](#) no Guia EC2 do usuário da Amazon.

Exemplo 4: executar uma instância e adicionar tags na criação

O exemplo `run-instances` a seguir adiciona uma tag com uma chave de `webserver` e um valor de `production` à instância. O comando também aplica uma tag com uma chave de `cost-center` e um valor de `cc123` a qualquer EBS volume criado (nesse caso, o volume raiz).

```
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}]'
```

Para obter um exemplo da saída de `run-instances`, consulte o Exemplo 1.

Exemplo 5: executar uma instância com dados do usuário

O exemplo `run-instances` a seguir passa dados do usuário em um arquivo chamado `my_script.txt` que contém um script de configuração para a sua instância. O script é executado na inicialização.

```
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
```

Para obter um exemplo da saída de `run-instances`, consulte o Exemplo 1.

Para obter mais informações sobre os dados do usuário da instância, consulte Como [trabalhar com dados do usuário da instância](#) no Guia EC2 do usuário da Amazon.

Exemplo 6: executar uma instância de desempenho expansível

O exemplo `run-instances` a seguir executa uma instância `t2.micro` com a opção de crédito `unlimited`. Ao executar uma instância T2, se você não especificar a `--credit-specification`, o padrão é a opção de crédito `standard`. Ao executar uma instância T3, o padrão é a opção de crédito `unlimited`.

```
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
```

Para obter um exemplo da saída de `run-instances`, consulte o Exemplo 1.

Para obter mais informações sobre instâncias de desempenho intermitentes, consulte Instâncias de [desempenho intermitentes](#) no Guia EC2 do usuário da Amazon.

- Para API obter detalhes, consulte [RunInstances](#) na Referência de AWS CLI Comandos.

Java

SDK para Java 2.x

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
/**
 * Runs an EC2 instance asynchronously.
 *
 * @param instanceType The instance type to use for the EC2 instance.
 * @param keyName The name of the key pair to associate with the EC2
instance.
 * @param groupName The name of the security group to associate with the EC2
instance.
 * @param amiId The ID of the Amazon Machine Image (AMI) to use for the EC2
instance.
 * @return A {@link CompletableFuture} that completes with the ID of the
started EC2 instance.
 * @throws RuntimeException If there is an error running the EC2 instance.
 */
public CompletableFuture<String> runInstanceAsync(String instanceType, String
keyName, String groupName, String amiId) {
    RunInstancesRequest runRequest = RunInstancesRequest.builder()
        .instanceType(instanceType)
        .keyName(keyName)
        .securityGroups(groupName)
        .maxCount(1)
        .minCount(1)
        .imageId(amiId)
        .build();

    CompletableFuture<RunInstancesResponse> responseFuture =
getAsyncClient().runInstances(runRequest);
    return responseFuture.thenCompose(response -> {
        String instanceIdVal = response.instances().get(0).instanceId();
    });
}
```



```

        System.out.println("Going to start an EC2 instance and use a waiter
to wait for it to be in running state");
        return getAsyncClient().waiter()
            .waitUntilInstanceExists(r -> r.instanceIds(instanceIdVal))
            .thenCompose(waitResponse -> getAsyncClient().waiter()
                .waitUntilInstanceRunning(r -> r.instanceIds(instanceIdVal))
                .thenApply(runningResponse -> instanceIdVal));
    }).exceptionally(throwable -> {
        // Handle any exceptions that occurred during the async call
        throw new RuntimeException("Failed to run EC2 instance: " +
            throwable.getMessage(), throwable);
    });
}

```

- Para API obter detalhes, consulte [RunInstances](#) em AWS SDK for Java 2.x API Referência.

JavaScript

SDK para JavaScript (v3)

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```

import { EC2Client, RunInstancesCommand } from "@aws-sdk/client-ec2";

/**
 * Create new EC2 instances.
 * @param {{
 *   keyName: string,
 *   securityGroupIds: string[],
 *   imageId: string,
 *   instanceType: import('@aws-sdk/client-ec2')._InstanceType,
 *   minCount?: number,
 *   maxCount?: number }} options
 */
export const main = async ({
    keyName,
    securityGroupIds,

```

```
    imageId,
    instanceType,
    minCount = "1",
    maxCount = "1",
  }) => {
    const client = new EC2Client({});
    minCount = Number.parseInt(minCount);
    maxCount = Number.parseInt(maxCount);
    const command = new RunInstancesCommand({
      // Your key pair name.
      KeyName: keyName,
      // Your security group.
      SecurityGroupIds: securityGroupIds,
      // An Amazon Machine Image (AMI). There are multiple ways to search for AMIs.
      // For more information, see:
      // https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/finding-an-ami.html
      ImageId: imageId,
      // An instance type describing the resources provided to your instance. There
      // are multiple
      // ways to search for instance types. For more information see:
      // https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/instance-
      // discovery.html
      InstanceType: instanceType,
      // Availability Zones have capacity limitations that may impact your ability
      // to launch instances.
      // The `RunInstances` operation will only succeed if it can allocate at least
      // the `MinCount` of instances.
      // However, EC2 will attempt to launch up to the `MaxCount` of instances,
      // even if the full request cannot be satisfied.
      // If you need a specific number of instances, use `MinCount` and `MaxCount`
      // set to the same value.
      // If you want to launch up to a certain number of instances, use `MaxCount`
      // and let EC2 provision as many as possible.
      // If you require a minimum number of instances, but do not want to exceed a
      // maximum, use both `MinCount` and `MaxCount`.
      MinCount: minCount,
      MaxCount: maxCount,
    });

    try {
      const { Instances } = await client.send(command);
      const instanceList = Instances.map(
        (instance) => `• ${instance.InstanceId}`,
      ).join("\n");
    }
  }
}
```

```
console.log(`Launched instances:\n${instanceList}`);
} catch (caught) {
  if (caught instanceof Error && caught.name === "ResourceCountExceeded") {
    console.warn(`${caught.message}`);
  } else {
    throw caught;
  }
}
};
```

- Para API obter detalhes, consulte [RunInstances](#) em AWS SDK for JavaScript API Referência.

Kotlin

SDK para Kotlin

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
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
}
}
```

- Para API obter detalhes, consulte a [RunInstances](#) referência AWS SDK do Kotlin API.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo inicia uma única instância do especificado AMI em EC2 -Classic ou um padrão VPC.

```
New-EC2Instance -ImageId ami-12345678 -MinCount 1 -MaxCount 1 -InstanceType
m3.medium -KeyName my-key-pair -SecurityGroup my-security-group
```

Exemplo 2: Este exemplo inicia uma única instância do especificado AMI em VPC a.

```
New-EC2Instance -ImageId ami-12345678 -MinCount 1 -MaxCount 1 -SubnetId
subnet-12345678 -InstanceType t2.micro -KeyName my-key-pair -SecurityGroupId
sg-12345678
```

Exemplo 3: Para adicionar um EBS volume ou um volume de armazenamento de instâncias, defina um mapeamento de dispositivos de blocos e adicione-o ao comando. Este exemplo adiciona um volume de armazenamento de instâncias.

```
$bdm = New-Object Amazon.EC2.Model.BlockDeviceMapping
$bdm.VirtualName = "ephemeral0"
$bdm.DeviceName = "/dev/sdf"
```

```
New-EC2Instance -ImageId ami-12345678 -BlockDeviceMapping $bdm ...
```

Exemplo 4: Para especificar um dos Windows atuais AMIs, obtenha seu AMI ID usando `Get-EC2ImageByName`. Este exemplo inicia uma instância da base atual AMI para o Windows Server 2016.

```
$ami = Get-EC2ImageByName WINDOWS_2016_BASE  
  
New-EC2Instance -ImageId $ami.ImageId ...
```

Exemplo 5: executa uma instância no ambiente de host dedicado especificado.

```
New-EC2Instance -ImageId ami-1a2b3c4d -InstanceType m4.large -KeyName my-key-pair  
-SecurityGroupId sg-1a2b3c4d -AvailabilityZone us-west-1a -Tenancy host -HostID  
h-1a2b3c4d5e6f1a2b3
```

Exemplo 6: Essa solicitação inicia duas instâncias e aplica uma tag com uma chave de servidor web e um valor de produção às instâncias. A solicitação também aplica uma tag com uma chave de centro de custos e um valor de cc123 aos volumes criados (nesse caso, o volume raiz de cada instância).

```
$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
```

- Para API obter detalhes, consulte [RunInstances](#) em Referência de AWS Tools for PowerShell cmdlet.

Python

SDK para Python (Boto3)

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
class EC2InstanceWrapper:
    """Encapsulates Amazon Elastic Compute Cloud (Amazon EC2) instance actions
    using the client interface."""

    def __init__(
        self, ec2_client: Any, instances: Optional[List[Dict[str, Any]]] = None
    ) -> None:
        """
        Initializes the EC2InstanceWrapper with an EC2 client and optional
        instances.

        :param ec2_client: A Boto3 Amazon EC2 client. This client provides low-
        level
                           access to AWS EC2 services.
        :param instances: A list of dictionaries representing Boto3 Instance
        objects. These are high-level objects that
                           wrap instance actions.
        """
        self.ec2_client = ec2_client
        self.instances = instances or []

    @classmethod
    def from_client(cls) -> "EC2InstanceWrapper":
        """
        Creates an EC2InstanceWrapper instance with a default EC2 client.

        :return: An instance of EC2InstanceWrapper initialized with the default
        EC2 client.
        """
        ec2_client = boto3.client("ec2")
        return cls(ec2_client)
```

```

def create(
    self,
    image_id: str,
    instance_type: str,
    key_pair_name: str,
    security_group_ids: Optional[List[str]] = None,
) -> List[Dict[str, Any]]:
    """
    Creates a new EC2 instance in the default VPC of the current account.

    The instance starts immediately after it is created.

    :param image_id: The ID of the Amazon Machine Image (AMI) to use for the
    instance.
    :param instance_type: The type of instance to create, such as 't2.micro'.
    :param key_pair_name: The name of the key pair to use for SSH access.
    :param security_group_ids: A list of security group IDs to associate with
    the instance.

                                If not specified, the default security group
    of the VPC is used.
    :return: A list of dictionaries representing Boto3 Instance objects
    representing the newly created instances.
    """
    try:
        instance_params = {
            "ImageId": image_id,
            "InstanceType": instance_type,
            "KeyName": key_pair_name,
        }
        if security_group_ids is not None:
            instance_params["SecurityGroupIds"] = security_group_ids

        response = self.ec2_client.run_instances(
            **instance_params, MinCount=1, MaxCount=1
        )
        instance = response["Instances"][0]
        self.instances.append(instance)
        waiter = self.ec2_client.get_waiter("instance_running")
        waiter.wait(InstanceIds=[instance["InstanceId"]])
    except ClientError as err:
        params_str = "\n\t".join(
            f"{key}: {value}" for key, value in instance_params.items()
        )

```

```
        logger.error(
            f"Failed to complete instance creation request.\nRequest details:
{params_str}"
        )
        error_code = err.response["Error"]["Code"]
        if error_code == "InstanceLimitExceeded":
            logger.error(
                (
                    f"Insufficient capacity for instance type
'{instance_type}'. "
                    "Terminate unused instances or contact AWS Support for a
limit increase."
                )
            )
        if error_code == "InsufficientInstanceCapacity":
            logger.error(
                (
                    f"Insufficient capacity for instance type
'{instance_type}'. "
                    "Select a different instance type or launch in a
different availability zone."
                )
            )
        raise
    return self.instances
```

- Para API obter detalhes, consulte a [RunInstances](#) Referência AWS SDK do Python (Boto3). API

Rust

SDK para Rust

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).


```
pub async fn create_instance<'a>(
    &self,
    image_id: &'a str,
    instance_type: InstanceType,
    key_pair: &'a KeyPairInfo,
    security_groups: Vec<&'a SecurityGroup>,
) -> Result<String, EC2Error> {
    let run_instances = self
        .client
        .run_instances()
        .image_id(image_id)
        .instance_type(instance_type)
        .key_name(
            key_pair
                .key_name()
                .ok_or_else(|| EC2Error::new("Missing key name when launching
instance"))?,
        )
        .set_security_group_ids(Some(
            security_groups
                .iter()
                .filter_map(|sg| sg.group_id.clone())
                .collect(),
        ))
        .min_count(1)
        .max_count(1)
        .send()
        .await?;

    if run_instances.instances().is_empty() {
        return Err(EC2Error::new("Failed to create instance"));
    }

    let instance_id = run_instances.instances()[0].instance_id().unwrap();
    let response = self
        .client
        .create_tags()
        .resources(instance_id)
        .tags(
            Tag::builder()
                .key("Name")
                .value("From SDK Examples")
                .build(),
        )
```

```

    )
    .send()
    .await;

    match response {
        Ok(_) => tracing::info!("Created {instance_id} and applied tags."),
        Err(err) => {
            tracing::info!("Error applying tags to {instance_id}: {err:?}");
            return Err(err.into());
        }
    }

    tracing::info!("Instance is created.");

    Ok(instance_id.to_string())
}

```

- Para API obter detalhes, consulte [RunInstances](#) a AWS SDKAPIreferência do Rust.

SAP ABAP

SDKpara SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```

" Create tags for resource created during instance launch. "
DATA lt_tagsspecifications TYPE /aws1/
c1_ec2tagsspecification=>tt_tagsspecificationlist.
DATA ls_tagsspecifications LIKE LINE OF lt_tagsspecifications.
ls_tagsspecifications = NEW /aws1/c1_ec2tagsspecification(
    iv_resourcetype = 'instance'
    it_tags = VALUE /aws1/c1_ec2tag=>tt_taglist(
        ( NEW /aws1/c1_ec2tag( iv_key = 'Name' iv_value = iv_tag_value ) )
    )
).
APPEND ls_tagsspecifications TO lt_tagsspecifications.

```

```

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.

```

- Para API obter detalhes, consulte [RunInstances AWS SDK](#) para SAP ABAP API referência.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **RunScheduledInstances** com um CLI

Os exemplos de código a seguir mostram como usar o `RunScheduledInstances`.

CLI

AWS CLI

Para iniciar uma instância programada

Este exemplo inicia a instância agendada especificada em um VPC.

Comando:

```
aws ec2 run-scheduled-instances --scheduled-instance-id sci-1234-1234-1234-1234-123456789012 --instance-count 1 --launch-specification file://launch-specification.json
```

Especificação de lançamento.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"
  }
}
```

Saída:

```
{
  "InstanceIdSet": [
    "i-1234567890abcdef0"
  ]
}
```

Este exemplo inicia a instância agendada especificada em EC2 -Classic.

Comando:

```
aws ec2 run-scheduled-instances --scheduled-instance-id sci-1234-1234-1234-1234-123456789012 --instance-count 1 --launch-specification file://launch-specification.json
```

Especificação de lançamento.json:

```
{
  "ImageId": "ami-12345678",
  "KeyName": "my-key-pair",
  "SecurityGroupIds": ["sg-12345678"],
  "InstanceType": "c4.large",
  "Placement": {
    "AvailabilityZone": "us-west-2b"
  }
  "IamInstanceProfile": {
    "Name": "my-iam-role"
  }
}
```

Saída:

```
{
  "InstanceIdSet": [
    "i-1234567890abcdef0"
  ]
}
```

- Para API obter detalhes, consulte [RunScheduledInstances](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo inicia a instância agendada especificada.

```
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
```

- Para API obter detalhes, consulte [RunScheduledInstances](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **StartInstances** com um AWS SDK ou CLI

Os exemplos de código a seguir mostram como usar o `StartInstances`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação no contexto no seguinte exemplo de código:

- [Aprender os conceitos básicos](#)

.NET

AWS SDK for .NET

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
/// <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)
{
    try
    {
        var request = new StartInstancesRequest
        {
            InstanceIds = new List<string> { ec2InstanceId },
        };

        await _amazonEC2.StartInstancesAsync(request);

        Console.WriteLine("Waiting for instance to start. ");
    }
}
```

```
        await WaitForInstanceState(ec2InstanceId, InstanceStateName.Running);
    }
    catch (AmazonEC2Exception ec2Exception)
    {
        if (ec2Exception.ErrorCode == "InvalidInstanceId")
        {
            _logger.LogError(
                $"InstanceId is invalid, unable to start.
{ec2Exception.Message}");
        }

        throw;
    }
    catch (Exception ex)
    {
        _logger.LogError(
            $"An error occurred while starting the instance.: {ex.Message}");
        throw;
    }
}

/// <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 in the specified state.
    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.WriteLine(". ");
    } while (!hasState);

    return hasState;
}

```

- Para API obter detalhes, consulte [StartInstances](#) em AWS SDK for .NET API Referência.

Bash

AWS CLI com script Bash

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```

#####
# 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 getopt 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)."
```

As funções utilitárias usadas neste exemplo.

```
#####
# 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
}
}
```

- Para API obter detalhes, consulte [StartInstances](#) na Referência de AWS CLI Comandos.

C++

SDK para C++

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
#!/ Start an Amazon Elastic Compute Cloud (Amazon EC2) instance.
/*!
 \param instanceID: An EC2 instance ID.
 \param clientConfiguration: AWS client configuration.
 \return bool: Function succeeded.
 */
bool AwsDoc::EC2::startInstance(const Aws::String &instanceId,
                               const Aws::Client::ClientConfiguration
&clientConfiguration) {
    Aws::EC2::EC2Client ec2Client(clientConfiguration);

    Aws::EC2::Model::StartInstancesRequest startRequest;
    startRequest.AddInstanceIds(instanceId);
    startRequest.SetDryRun(true);

    Aws::EC2::Model::StartInstancesOutcome dryRunOutcome =
ec2Client.StartInstances(startRequest);
    if (dryRunOutcome.IsSuccess()) {
        std::cerr
            << "Failed dry run to start instance. A dry run should trigger an
error."
            << std::endl;
        return false;
    } else if (dryRunOutcome.GetError().GetErrorType() !=
        Aws::EC2::EC2Errors::DRY_RUN_OPERATION) {
        std::cout << "Failed dry run to start instance " << instanceId << ": "
            << dryRunOutcome.GetError().GetMessage() << std::endl;
        return false;
    }
}
```

```
    }

    startRequest.SetDryRun(false);
    Aws::EC2::Model::StartInstancesOutcome startInstancesOutcome =
    ec2Client.StartInstances(startRequest);

    if (!startInstancesOutcome.IsSuccess()) {
        std::cout << "Failed to start instance " << instanceId << ": " <<
            startInstancesOutcome.GetError().GetMessage() << std::endl;
    } else {
        std::cout << "Successfully started instance " << instanceId <<
            std::endl;
    }

    return startInstancesOutcome.IsSuccess();
}
```

- Para API obter detalhes, consulte [StartInstances](#) em AWS SDK for C++ API Referência.

CLI

AWS CLI

Para iniciar uma EC2 instância da Amazon

Este exemplo inicia a instância especificada com EBS suporte da Amazon.

Comando:

```
aws ec2 start-instances --instance-ids i-1234567890abcdef0
```

Saída:

```
{
  "StartingInstances": [
    {
      "InstanceId": "i-1234567890abcdef0",
      "CurrentState": {
        "Code": 0,
        "Name": "pending"
      }
    },
  ],
}
```

```
        "PreviousState": {
            "Code": 80,
            "Name": "stopped"
        }
    }
]
```

Para obter mais informações, consulte Interromper e iniciar sua instância no Guia do usuário do Amazon Elastic Compute Cloud.

- Para API obter detalhes, consulte [StartInstances](#) na Referência de AWS CLI Comandos.

Java

SDK para Java 2.x

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
/**
 * Starts an Amazon EC2 instance asynchronously and waits until it is in the
 "running" state.
 *
 * @param instanceId the ID of the instance to start
 * @return a {@link CompletableFuture} that completes when the instance has
 been started and is in the "running" state, or exceptionally if an error occurs
 */
public CompletableFuture<Void> startInstanceAsync(String instanceId) {
    StartInstancesRequest startRequest = StartInstancesRequest.builder()
        .instanceIds(instanceId)
        .build();

    Ec2AsyncWaiter ec2Waiter = Ec2AsyncWaiter.builder()
        .client(getAsyncClient())
        .build();

    DescribeInstancesRequest describeRequest =
DescribeInstancesRequest.builder()
```

```
        .instanceIds(instanceId)
        .build();

    logger.info("Starting instance " + instanceId + " and waiting for it to
run.");
    CompletableFuture<Void> resultFuture = new CompletableFuture<>();
    return getAsyncClient().startInstances(startRequest)
        .thenCompose(response ->
            ec2Waiter.waitForInstanceRunning(describeRequest)
        )
        .thenAccept(waiterResponse -> {
            logger.info("Successfully started instance " + instanceId);
            resultFuture.complete(null);
        })
        .exceptionally(throwable -> {
            resultFuture.completeExceptionally(new RuntimeException("Failed
to start instance: " + throwable.getMessage(), throwable));
            return null;
        });
    }
```

- Para API obter detalhes, consulte [StartInstances](#) em AWS SDK for Java 2.x API Referência.

JavaScript

SDK para JavaScript (v3)

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
import { EC2Client, StartInstancesCommand } from "@aws-sdk/client-ec2";
import { fileURLToPath } from "node:url";
import { parseArgs } from "node:util";

/**
 * Starts an Amazon EBS-backed instance that you've previously stopped.
 * @param {{ instanceIds }} options
 */
```

```
export const main = async ({ instanceIds }) => {
  const client = new EC2Client({});
  const command = new StartInstancesCommand({
    InstanceIds: instanceIds,
  });

  try {
    const { StartingInstances } = await client.send(command);
    const instanceIdList = StartingInstances.map(
      (instance) => ` • ${instance.InstanceId}`,
    );
    console.log("Starting instances:");
    console.log(instanceIdList.join("\n"));
  } catch (caught) {
    if (
      caught instanceof Error &&
      caught.name === "InvalidInstanceID.NotFound"
    ) {
      console.warn(`${caught.message}`);
    } else {
      throw caught;
    }
  }
};
```

- Para API obter detalhes, consulte [StartInstances](#) em AWS SDK for JavaScript API Referência.

Kotlin

SDK para Kotlin

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
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.waitForInstanceRunning {
                // suspend call
                instanceIds = listOf(instanceId)
            }
            println("Successfully started instance $instanceId")
        }
    }
}

```

- Para API obter detalhes, consulte a [StartInstances](#) referência AWS SDK do Kotlin API.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Esse exemplo inicia a instância especificada.

```
Start-EC2Instance -InstanceId i-12345678
```

Saída:

CurrentState	InstanceId	PreviousState
-----	-----	-----
Amazon.EC2.Model.InstanceState	i-12345678	Amazon.EC2.Model.InstanceState

Exemplo 2: Este exemplo inicia as instâncias especificadas.

```
@("i-12345678", "i-76543210") | Start-EC2Instance
```

Exemplo 3: Este exemplo inicia o conjunto de instâncias que estão atualmente paradas. Os objetos Instance retornados por Get-EC2Instance são canalizados para Start-EC2Instance. A sintaxe usada neste exemplo requer a PowerShell versão 3 ou superior.


```
(Get-EC2Instance -Filter @{ Name="instance-state-name";
  Values="stopped"}).Instances | Start-EC2Instance
```

Exemplo 4: Com a PowerShell versão 2, você deve usar `New-Object` para criar o filtro para o parâmetro `Filter`.

```
$filter = New-Object Amazon.EC2.Model.Filter
$filter.Name = "instance-state-name"
$filter.Values = "stopped"

(Get-EC2Instance -Filter $filter).Instances | Start-EC2Instance
```

- Para API obter detalhes, consulte [StartInstances](#) em Referência de AWS Tools for PowerShell cmdlet.

Python

SDK para Python (Boto3)

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
class EC2InstanceWrapper:
    """Encapsulates Amazon Elastic Compute Cloud (Amazon EC2) instance actions
    using the client interface."""

    def __init__(
        self, ec2_client: Any, instances: Optional[List[Dict[str, Any]]] = None
    ) -> None:
        """
        Initializes the EC2InstanceWrapper with an EC2 client and optional
        instances.

        :param ec2_client: A Boto3 Amazon EC2 client. This client provides low-
        level
                           access to AWS EC2 services.
```

```

        :param instances: A list of dictionaries representing Boto3 Instance
objects. These are high-level objects that
                        wrap instance actions.
        """
        self.ec2_client = ec2_client
        self.instances = instances or []

    @classmethod
    def from_client(cls) -> "EC2InstanceWrapper":
        """
        Creates an EC2InstanceWrapper instance with a default EC2 client.

        :return: An instance of EC2InstanceWrapper initialized with the default
EC2 client.
        """
        ec2_client = boto3.client("ec2")
        return cls(ec2_client)

    def start(self) -> Optional[Dict[str, Any]]:
        """
        Starts instances and waits for them to be in a running state.

        :return: The response to the start request.
        """
        if not self.instances:
            logger.info("No instances to start.")
            return None

        instance_ids = [instance["InstanceId"] for instance in self.instances]
        try:
            start_response =
self.ec2_client.start_instances(InstanceIds=instance_ids)
            waiter = self.ec2_client.get_waiter("instance_running")
            waiter.wait(InstanceIds=instance_ids)
            return start_response
        except ClientError as err:
            logger.error(
                f"Failed to start instance(s): {','.join(map(str,
instance_ids))}")
            )
            error_code = err.response["Error"]["Code"]
            if error_code == "IncorrectInstanceState":
                logger.error(

```

```

        "Couldn't start instance(s) because they are in an incorrect
state. "
        "Ensure the instances are in a stopped state before starting
them."
    )
    raise

```

- Para API obter detalhes, consulte a [StartInstances](#) Referência AWS SDK do Python (Boto3). API

Ruby

SDK para Ruby

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```

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.'
true
rescue StandardError => e
  puts "Error starting instance: #{e.message}"
  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)... '
return if instance_started?(ec2_client, instance_id)

puts 'Could not start instance.'
end

run_me if $PROGRAM_NAME == __FILE__
```

- Para API obter detalhes, consulte [StartInstances](#) em AWS SDK for Ruby API Referência.

Rust

SDK para Rust

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

Inicie uma EC2 instância pelo ID da instância.

```
pub async fn start_instance(&self, instance_id: &str) -> Result<(), EC2Error>
{
  tracing::info!("Starting instance {instance_id}");

  self.client
    .start_instances()
    .instance_ids(instance_id)
    .send()
    .await?;

  tracing::info!("Started instance.");
}
```

```
    Ok(())
}
```

Aguarde até que uma instância esteja pronta e com o status ok, usando os WaitersAPI. Usar os Waiters API requer `use aws_sdk_ec2: :client: :Waiters` no arquivo rust.

```
/// Wait for an instance to be ready and status ok (default wait 60 seconds)
pub async fn wait_for_instance_ready(
    &self,
    instance_id: &str,
    duration: Option<Duration>,
) -> Result<(), EC2Error> {
    self.client
        .wait_until_instance_status_ok()
        .instance_ids(instance_id)
        .wait(duration.unwrap_or(Duration::from_secs(60)))
        .await
        .map_err(|err| match err {
            WaiterError::ExceededMaxWait(exceeded) => EC2Error(format!(
                "Exceeded max time ({}s) waiting for instance to start.",
                exceeded.max_wait().as_secs()
            )),
            _ => EC2Error::from(err),
        })?;
    Ok(())
}
```

- Para API obter detalhes, consulte [StartInstances](#) a AWS SDKAPIreferência do Rust.

SAP ABAP

SDKpara SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```

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.
  ENDTRY.

```

- Para API obter detalhes, consulte [StartInstances AWS SDK](#) para SAP ABAP API referência.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **StopInstances** com um AWS SDK ou CLI

Os exemplos de código a seguir mostram como usar o `StopInstances`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação no contexto no seguinte exemplo de código:

- [Aprender os conceitos básicos](#)

.NET

AWS SDK for .NET

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
/// <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)
{
    try
    {
        var request = new StopInstancesRequest
        {
            InstanceIds = new List<string> { ec2InstanceId },
        };

        await _amazonEC2.StopInstancesAsync(request);
        Console.WriteLine("Waiting for the instance to stop.");
        await WaitForInstanceState(ec2InstanceId, InstanceStateName.Stopped);
    }
}
```



```
        Console.WriteLine("\nThe instance has stopped.");
    }
    catch (AmazonEC2Exception ec2Exception)
    {
        if (ec2Exception.ErrorCode == "InvalidInstanceId")
        {
            _logger.LogError(
                $"InstanceId is invalid, unable to stop.
{ec2Exception.Message}");
        }

        throw;
    }
    catch (Exception ex)
    {
        _logger.LogError(
            $"An error occurred while stopping the instance.: {ex.Message}");
        throw;
    }
}

/// <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 in the specified state.
    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.WriteLine(". ");
    } while (!hasState);

    return hasState;
}

```

- Para API obter detalhes, consulte [StopInstances](#) em AWS SDK for .NET API Referência.

Bash

AWS CLI com script Bash

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```

#####
# 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 getopt 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
}
```

As funções utilitárias usadas neste exemplo.

```
#####
# 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
}
```

- Para API obter detalhes, consulte [StopInstances](#) na Referência de AWS CLI Comandos.

C++

SDK para C++

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
#!/ Stop an EC2 instance.
/*!
 \param instanceID: An EC2 instance ID.
 \param clientConfiguration: AWS client configuration.
 \return bool: Function succeeded.
 */
bool AwsDoc::EC2::stopInstance(const Aws::String &instanceId,
                               const Aws::Client::ClientConfiguration
                               &clientConfiguration) {
    Aws::EC2::EC2Client ec2Client(clientConfiguration);
    Aws::EC2::Model::StopInstancesRequest request;
    request.AddInstanceIds(instanceId);
    request.SetDryRun(true);

    Aws::EC2::Model::StopInstancesOutcome dryRunOutcome =
    ec2Client.StopInstances(request);
    if (dryRunOutcome.IsSuccess()) {
        std::cerr
            << "Failed dry run to stop instance. A dry run should trigger an
            error."
            << std::endl;
        return false;
    } else if (dryRunOutcome.GetError().GetErrorType() !=
                Aws::EC2::EC2Errors::DRY_RUN_OPERATION) {
        std::cout << "Failed dry run to stop instance " << instanceId << ": "
            << dryRunOutcome.GetError().GetMessage() << std::endl;
        return false;
    }
}
```

```

    request.SetDryRun(false);
    Aws::EC2::Model::StopInstancesOutcome 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;
    }

    return outcome.IsSuccess();
}

void PrintUsage() {
    std::cout << "Usage: run_start_stop_instance <instance_id> <start|stop>" <<
        std::endl;
}

```

- Para API obter detalhes, consulte [StopInstances](#) em AWS SDK for C++ API Referência.

CLI

AWS CLI

Exemplo 1: Para parar uma EC2 instância da Amazon

O stop-instances exemplo a seguir interrompe a instância especificada EBS com suporte da Amazon.

```

aws ec2 stop-instances \
  --instance-ids i-1234567890abcdef0

```

Saída:

```

{
  "StoppingInstances": [
    {
      "InstanceId": "i-1234567890abcdef0",
      "CurrentState": {

```

```

        "Code": 64,
        "Name": "stopping"
    },
    "PreviousState": {
        "Code": 16,
        "Name": "running"
    }
}
]
}

```

Para obter mais informações, consulte [Interromper e iniciar sua instância](#) no Guia do usuário do Amazon Elastic Compute Cloud.

Exemplo 2: Para hibernar uma instância da Amazon EC2

O `stop-instances` exemplo a seguir hiberna a instância EBS apoiada pela Amazon se a instância estiver habilitada para hibernação e atender aos pré-requisitos de hibernação. Depois de colocar a instância em hibernação, ela é interrompida.

```

aws ec2 stop-instances \
  --instance-ids i-1234567890abcdef0 \
  --hibernate

```

Saída:

```

{
  "StoppingInstances": [
    {
      "CurrentState": {
        "Code": 64,
        "Name": "stopping"
      },
      "InstanceId": "i-1234567890abcdef0",
      "PreviousState": {
        "Code": 16,
        "Name": "running"
      }
    }
  ]
}

```

Para obter mais informações, consulte [Colocar em hibernação uma instância sob demanda do Linux](#) no Guia do usuário do Amazon Elastic Cloud Compute.

- Para API obter detalhes, consulte [StopInstances](#) na Referência de AWS CLI Comandos.

Java

SDK para Java 2.x

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
/**
 * Stops the EC2 instance with the specified ID asynchronously and waits for
 the instance to stop.
 *
 * @param instanceId the ID of the EC2 instance to stop
 * @return a {@link CompletableFuture} that completes when the instance has
 been stopped, or exceptionally if an error occurs
 */
public CompletableFuture<Void> stopInstanceAsync(String instanceId) {
    StopInstancesRequest stopRequest = StopInstancesRequest.builder()
        .instanceIds(instanceId)
        .build();

    DescribeInstancesRequest describeRequest =
DescribeInstancesRequest.builder()
        .instanceIds(instanceId)
        .build();

    Ec2AsyncWaiter ec2Waiter = Ec2AsyncWaiter.builder()
        .client(getAsyncClient())
        .build();

    CompletableFuture<Void> resultFuture = new CompletableFuture<>();
    logger.info("Stopping instance " + instanceId + " and waiting for it to
stop.");
    getAsyncClient().stopInstances(stopRequest)
        .thenCompose(response -> {
```



```

        if (response.stoppingInstances().isEmpty()) {
            return CompletableFuture.failedFuture(new
RuntimeException("No instances were stopped. Please check the instance ID: " +
instanceId));
        }
        return ec2Waiter.waitUntilInstanceStopped(describeRequest);
    })
    .thenAccept(waiterResponse -> {
        logger.info("Successfully stopped instance " + instanceId);
        resultFuture.complete(null);
    })
    .exceptionally(throwable -> {
        logger.error("Failed to stop instance " + instanceId + ": " +
throwable.getMessage(), throwable);
        resultFuture.completeExceptionally(new RuntimeException("Failed
to stop instance: " + throwable.getMessage(), throwable));
        return null;
    });
    });

    return resultFuture;
}

```

- Para API obter detalhes, consulte [StopInstances](#) em AWS SDK for Java 2.x API Referência.

JavaScript

SDK para JavaScript (v3)

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```

import { EC2Client, StopInstancesCommand } from "@aws-sdk/client-ec2";
import { fileURLToPath } from "node:url";
import { parseArgs } from "node:util";

/**
 * Stop one or more EC2 instances.
 * @param {{ instanceIds: string[] }} options

```

```
*/
export const main = async ({ instanceIds }) => {
  const client = new EC2Client({});
  const command = new StopInstancesCommand({
    InstanceIds: instanceIds,
  });

  try {
    const { StoppingInstances } = await client.send(command);
    const instanceIdList = StoppingInstances.map(
      (instance) => ` • ${instance.InstanceId}`,
    );
    console.log("Stopping instances:");
    console.log(instanceIdList.join("\n"));
  } catch (caught) {
    if (
      caught instanceof Error &&
      caught.name === "InvalidInstanceID.NotFound"
    ) {
      console.warn(`${caught.message}`);
    } else {
      throw caught;
    }
  }
};
```

- Para API obter detalhes, consulte [StopInstances](#) em AWS SDK for JavaScript API Referência.

Kotlin

SDK para Kotlin

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
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")
}
}
```

- Para API obter detalhes, consulte a [StopInstances](#) referência AWS SDK do Kotlin API.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Esse exemplo interrompe a instância especificada.

```
Stop-EC2Instance -InstanceId i-12345678
```

Saída:

CurrentState	InstanceId	PreviousState
-----	-----	-----
Amazon.EC2.Model.InstanceState	i-12345678	Amazon.EC2.Model.InstanceState

- Para API obter detalhes, consulte [StopInstances](#) em Referência de AWS Tools for PowerShell cmdlet.

Python

SDK para Python (Boto3)

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
class EC2InstanceWrapper:
    """Encapsulates Amazon Elastic Compute Cloud (Amazon EC2) instance actions
    using the client interface."""

    def __init__(
        self, ec2_client: Any, instances: Optional[List[Dict[str, Any]]] = None
    ) -> None:
        """
        Initializes the EC2InstanceWrapper with an EC2 client and optional
        instances.

        :param ec2_client: A Boto3 Amazon EC2 client. This client provides low-
        level
                           access to AWS EC2 services.
        :param instances: A list of dictionaries representing Boto3 Instance
        objects. These are high-level objects that
                           wrap instance actions.
        """
        self.ec2_client = ec2_client
        self.instances = instances or []

    @classmethod
    def from_client(cls) -> "EC2InstanceWrapper":
        """
        Creates an EC2InstanceWrapper instance with a default EC2 client.

        :return: An instance of EC2InstanceWrapper initialized with the default
        EC2 client.
        """
        ec2_client = boto3.client("ec2")
        return cls(ec2_client)
```

```
def stop(self) -> Optional[Dict[str, Any]]:
    """
    Stops instances and waits for them to be in a stopped state.

    :return: The response to the stop request, or None if there are no
instances to stop.
    """
    if not self.instances:
        logger.info("No instances to stop.")
        return None

    instance_ids = [instance["InstanceId"] for instance in self.instances]
    try:
        # Attempt to stop the instances
        stop_response =
self.ec2_client.stop_instances(InstanceIds=instance_ids)
        waiter = self.ec2_client.get_waiter("instance_stopped")
        waiter.wait(InstanceIds=instance_ids)
    except ClientError as err:
        logger.error(
            f"Failed to stop instance(s): {','.join(map(str, instance_ids))}"
        )
        error_code = err.response["Error"]["Code"]
        if error_code == "IncorrectInstanceState":
            logger.error(
                "Couldn't stop instance(s) because they are in an incorrect
state. "
                "Ensure the instances are in a running state before stopping
them."
            )
            raise
    return stop_response
```

- Para API obter detalhes, consulte a [StopInstances](#) Referência AWS SDK do Python (Boto3).
API

Ruby

SDK para Ruby

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
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
end
```

```
end

ec2_client.stop_instances(instance_ids: [instance_id])
ec2_client.wait_until(:instance_stopped, instance_ids: [instance_id])
puts 'Instance stopped.'
true
rescue StandardError => e
  puts "Error stopping instance: #{e.message}"
  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)... '
  return if instance_stopped?(ec2_client, instance_id)

  puts 'Could not stop instance.'
end

run_me if $PROGRAM_NAME == __FILE__
```

- Para API obter detalhes, consulte [StopInstances](#) em AWS SDK for Ruby API Referência.

Rust

SDK para Rust

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
pub async fn stop_instance(&self, instance_id: &str) -> Result<(), EC2Error>
{
    tracing::info!("Stopping instance {instance_id}");

    self.client
        .stop_instances()
        .instance_ids(instance_id)
        .send()
        .await?;

    self.wait_for_instance_stopped(instance_id, None).await?;

    tracing::info!("Stopped instance.");

    Ok(())
}
```

Aguarde até que uma instância esteja no estado interrompido, usando os Waiters API. Usar os Waiters API requer `use aws_sdk_ec2: :client: :Waiters` no arquivo rust.

```
pub async fn stop_instance(&self, instance_id: &str) -> Result<(), EC2Error>
{
    tracing::info!("Stopping instance {instance_id}");

    self.client
```



```

        .stop_instances()
        .instance_ids(instance_id)
        .send()
        .await?;

self.wait_for_instance_stopped(instance_id, None).await?;

tracing::info!("Stopped instance.");

Ok(())
}

```

- Para API obter detalhes, consulte [StopInstances](#) a AWS SDKAPIreferência do Rust.

SAP ABAP

SDKpara SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```

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.
  ENDMETHOD.
ENDTRY.

```

- Para API obter detalhes, consulte [StopInstances AWS SDK](#) para SAP ABAP API referência.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **TerminateInstances** com um AWS SDK ou CLI

Os exemplos de código a seguir mostram como usar o `TerminateInstances`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação no contexto no seguinte exemplo de código:

- [Aprender os conceitos básicos](#)

.NET

AWS SDK for .NET

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
/// <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)
{
    try
    {
        var request = new TerminateInstancesRequest
        {
            InstanceIds = new List<string> { ec2InstanceId }
        };

        var response = await _amazonEC2.TerminateInstancesAsync(request);
        Console.WriteLine("Waiting for the instance to terminate.");
        await WaitForInstanceState(ec2InstanceId,
InstanceStateName.Terminated);

        Console.WriteLine($"\\nThe instance {ec2InstanceId} has been
terminated.");
        return response.TerminatingInstances;
    }
    catch (AmazonEC2Exception ec2Exception)
    {
        if (ec2Exception.ErrorCode == "InvalidInstanceId")
        {
            _logger.LogError(
                $"InstanceId is invalid, unable to terminate.
{ec2Exception.Message}");
        }
    }
}
```

```
        }

        throw;
    }
    catch (Exception ex)
    {
        _logger.LogError(
            $"An error occurred while terminating the instance.:
{ex.Message}");
        throw;
    }
}

/// <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 in the specified state.
    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.WriteLine(". ");
    } while (!hasState);

    return hasState;
}
```

- Para API obter detalhes, consulte [TerminateInstances](#) em AWS SDK for .NET API Referência.

Bash

AWS CLI com script Bash

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
#####
# 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 getopt 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
}

```

As funções utilitárias usadas neste exemplo.

```
#####
```

```
# 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
}
```

- Para API obter detalhes, consulte [TerminateInstances](#) na Referência de AWS CLI Comandos.

C++

SDK para C++

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
#!/ Terminate an Amazon Elastic Compute Cloud (Amazon EC2) instance.
/*!
  \param instanceID: An EC2 instance ID.
  \param clientConfiguration: AWS client configuration.
  \return bool: Function succeeded.
 */
bool AwsDoc::EC2::terminateInstances(const Aws::String &instanceID,
                                     const Aws::Client::ClientConfiguration
&clientConfiguration) {
    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;
    }

    return outcome.IsSuccess();
}
```


- Para API obter detalhes, consulte [TerminateInstances](#) em AWS SDK for C++ API Referência.

CLI

AWS CLI

Para encerrar uma instância da Amazon EC2

Este exemplo encerra a instância especificada.

Comando:

```
aws ec2 terminate-instances --instance-ids i-1234567890abcdef0
```

Saída:


```
{
  "TerminatingInstances": [
    {
      "InstanceId": "i-1234567890abcdef0",
      "CurrentState": {
        "Code": 32,
        "Name": "shutting-down"
      },
      "PreviousState": {
        "Code": 16,
        "Name": "running"
      }
    }
  ]
}
```

Para obter mais informações, consulte Como usar EC2 instâncias da Amazon no Guia do usuário da interface de linha de AWS comando.

- Para API obter detalhes, consulte [TerminateInstances](#) na Referência de AWS CLI Comandos.

Java

SDK para Java 2.x

 Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
/**
 * Terminates an EC2 instance asynchronously and waits for it to reach the
 * terminated state.
 *
 * @param instanceId the ID of the EC2 instance to terminate
 * @return a {@link CompletableFuture} that completes when the instance has
 * been terminated
 * @throws RuntimeException if there is no response from the AWS SDK or if
 * there is a failure during the termination process
 */
public CompletableFuture<Object> terminateEC2Async(String instanceId) {
    TerminateInstancesRequest terminateRequest =
    TerminateInstancesRequest.builder()
        .instanceIds(instanceId)
        .build();

    CompletableFuture<TerminateInstancesResponse> responseFuture =
    getAsyncClient().terminateInstances(terminateRequest);
    return responseFuture.thenCompose(terminateResponse -> {
        if (terminateResponse == null) {
            throw new RuntimeException("No response received for terminating
            instance " + instanceId);
        }
        System.out.println("Going to terminate an EC2 instance and use a
        waiter to wait for it to be in terminated state");
        return getAsyncClient().waiter()
            .waitUntilInstanceTerminated(r -> r.instanceIds(instanceId))
            .thenApply(waiterResponse -> null);
    }).exceptionally(throwable -> {
        // Handle any exceptions that occurred during the async call
        throw new RuntimeException("Failed to terminate EC2 instance: " +
        throwable.getMessage(), throwable);
    });
}
```

```
});  
}
```

- Para API obter detalhes, consulte [TerminateInstances](#) em AWS SDK for Java 2.x API Referência.

JavaScript

SDK para JavaScript (v3)

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
import { EC2Client, TerminateInstancesCommand } from "@aws-sdk/client-ec2";  
import { fileURLToPath } from "node:url";  
import { parseArgs } from "node:util";  
  
/**  
 * Terminate one or more EC2 instances.  
 * @param {{ instanceIds: string[] }} options  
 */  
export const main = async ({ instanceIds }) => {  
  const client = new EC2Client({});  
  const command = new TerminateInstancesCommand({  
    InstanceIds: instanceIds,  
  });  
  
  try {  
    const { TerminatingInstances } = await client.send(command);  
    const instanceList = TerminatingInstances.map(  
      (instance) => ` • ${instance.InstanceId}`,  
    );  
    console.log("Terminating instances:");  
    console.log(instanceList.join("\n"));  
  } catch (caught) {  
    if (  
      caught instanceof Error &&
```

```
        caught.name === "InvalidInstanceID.NotFound"
    ) {
        console.warn(`${caught.message}`);
    } else {
        throw caught;
    }
}
};
```

- Para API obter detalhes, consulte [TerminateInstances](#) em AWS SDK for JavaScript API Referência.

Kotlin

SDK para Kotlin

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
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}")
        }
    }
}
```

- Para API obter detalhes, consulte a [TerminateInstances](#) referência AWS SDK do Kotlin API.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Esse exemplo encerra a instância especificada (a instância pode estar em execução ou no estado “interrompido”). O cmdlet solicitará a confirmação antes de continuar; use a opção `-Force` para suprimir a solicitação.

```
Remove-EC2Instance -InstanceId i-12345678
```

Saída:

CurrentState	InstanceId	PreviousState
-----	-----	-----
Amazon.EC2.Model.InstanceState	i-12345678	Amazon.EC2.Model.InstanceState

- Para API obter detalhes, consulte [TerminateInstances](#) em Referência de AWS Tools for PowerShell cmdlet.

Python

SDK para Python (Boto3)

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
class EC2InstanceWrapper:
    """Encapsulates Amazon Elastic Compute Cloud (Amazon EC2) instance actions
    using the client interface."""

    def __init__(
        self, ec2_client: Any, instances: Optional[List[Dict[str, Any]]] = None
    ) -> None:
        """
        Initializes the EC2InstanceWrapper with an EC2 client and optional
        instances.
```

```

        :param ec2_client: A Boto3 Amazon EC2 client. This client provides low-
level
        access to AWS EC2 services.
        :param instances: A list of dictionaries representing Boto3 Instance
objects. These are high-level objects that
        wrap instance actions.
        """
        self.ec2_client = ec2_client
        self.instances = instances or []

    @classmethod
    def from_client(cls) -> "EC2InstanceWrapper":
        """
        Creates an EC2InstanceWrapper instance with a default EC2 client.

        :return: An instance of EC2InstanceWrapper initialized with the default
EC2 client.
        """
        ec2_client = boto3.client("ec2")
        return cls(ec2_client)

    def terminate(self) -> None:
        """
        Terminates instances and waits for them to reach the terminated state.
        """
        if not self.instances:
            logger.info("No instances to terminate.")
            return

        instance_ids = [instance["InstanceId"] for instance in self.instances]
        try:
            self.ec2_client.terminate_instances(InstanceIds=instance_ids)
            waiter = self.ec2_client.get_waiter("instance_terminated")
            waiter.wait(InstanceIds=instance_ids)
            self.instances.clear()
            for instance_id in instance_ids:
                print(f"• Instance ID: {instance_id}\n" f"• Action: Terminated")

        except ClientError as err:
            logger.error(
                f"Failed instance termination details:\n\t{str(self.instances)}"
            )
            error_code = err.response["Error"]["Code"]

```

```
if error_code == "InvalidInstanceID.NotFound":
    logger.error(
        "One or more instance IDs do not exist. "
        "Please verify the instance IDs and try again."
    )
    raise
```

- Para API obter detalhes, consulte a [TerminateInstances](#) Referência AWS SDK do Python (Boto3). API

Ruby

SDK para Ruby

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
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.'
  true
rescue StandardError => e
  puts "Error terminating instance: #{e.message}"
  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)... '
  return if instance_terminated?(ec2_client, instance_id)
end
```



```
puts 'Could not terminate instance.'
end

run_me if $PROGRAM_NAME == __FILE__
```

- Para API obter detalhes, consulte [TerminateInstances](#) em AWS SDK for Ruby API Referência.

Rust

SDK para Rust

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
pub async fn delete_instance(&self, instance_id: &str) -> Result<(),
EC2Error> {
    tracing::info!("Deleting instance with id {instance_id}");
    self.stop_instance(instance_id).await?;
    self.client
        .terminate_instances()
        .instance_ids(instance_id)
        .send()
        .await?;
    self.wait_for_instance_terminated(instance_id).await?;
    tracing::info!("Terminated instance with id {instance_id}");
    Ok(())
}
```

Aguarde até que uma instância esteja no estado encerrado, usando os API Waiters. Usar os Waiters API requer `use aws_sdk_ec2: :client: :Waiters` no arquivo rust.

```
async fn wait_for_instance_terminated(&self, instance_id: &str) -> Result<(),
EC2Error> {
    self.client
```

```

        .wait_until_instance_terminated()
        .instance_ids(instance_id)
        .wait(Duration::from_secs(60))
        .await
        .map_err(|err| match err {
            WaiterError::ExceededMaxWait(exceeded) => EC2Error(format!(
                "Exceeded max time ({}s) waiting for instance to terminate.",
                exceeded.max_wait().as_secs(),
            )),
            _ => EC2Error::from(err),
        })?;
    Ok(())
}

```

- Para API obter detalhes, consulte [TerminateInstances](#) na AWS SDK API referência do Rust.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use `UnassignPrivateIpAddresses` com um CLI

Os exemplos de código a seguir mostram como usar o `UnassignPrivateIpAddresses`.

CLI

AWS CLI

Para cancelar a atribuição de um endereço IP privado secundário de uma interface de rede

Este exemplo cancela a atribuição do endereço IP privado especificado da interface de rede especificada. Se o comando for bem-sucedido, nenhuma saída será retornada.

Comando:

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

- Para API obter detalhes, consulte [UnassignPrivateIpAddresses](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo cancela a atribuição do endereço IP privado especificado da interface de rede especificada.

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

- Para API obter detalhes, consulte [UnassignPrivateIpAddresses](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use **UnmonitorInstances** com um AWS SDK ou CLI

Os exemplos de código a seguir mostram como usar o `UnmonitorInstances`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação no contexto no seguinte exemplo de código:

- [Aprender os conceitos básicos](#)

C++

SDK para C++

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
//! Disable monitoring for an EC2 instance.  
/*!  
  \param instanceId: An EC2 instance ID.  
  \param clientConfiguration: AWS client configuration.
```

```

    \return bool: Function succeeded.
    */
bool AwsDoc::EC2::disableMonitoring(const Aws::String &instanceId,
                                   const Aws::Client::ClientConfiguration
                                   &clientConfiguration) {
    Aws::EC2::EC2Client ec2Client(clientConfiguration);
    Aws::EC2::Model::UnmonitorInstancesRequest unrequest;
    unrequest.AddInstanceIds(instanceId);
    unrequest.SetDryRun(true);

    Aws::EC2::Model::UnmonitorInstancesOutcome dryRunOutcome =
ec2Client.UnmonitorInstances(unrequest);
    if (dryRunOutcome.IsSuccess()) {
        std::cerr
            << "Failed dry run to disable monitoring on instance. A dry run
should trigger an error."
            <<
            std::endl;
        return false;
    } else if (dryRunOutcome.GetError().GetErrorType() !=
                Aws::EC2::EC2Errors::DRY_RUN_OPERATION) {
        std::cout << "Failed dry run to disable monitoring on instance " <<
            instanceId << ": " << dryRunOutcome.GetError().GetMessage() <<
            std::endl;
        return false;
    }

    unrequest.SetDryRun(false);
    Aws::EC2::Model::UnmonitorInstancesOutcome 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;
    }

    return unmonitorInstancesOutcome.IsSuccess();
}

```

- Para API obter detalhes, consulte [UnmonitorInstances](#) em AWS SDK for C++ API Referência.

CLI

AWS CLI

Para desabilitar o monitoramento detalhado de uma instância

Este exemplo de comando desabilita o monitoramento detalhado da instância especificada.

Comando:

```
aws ec2 unmonitor-instances --instance-ids i-1234567890abcdef0
```

Saída:

```
{
  "InstanceMonitorings": [
    {
      "InstanceId": "i-1234567890abcdef0",
      "Monitoring": {
        "State": "disabling"
      }
    }
  ]
}
```

- Para API obter detalhes, consulte [UnmonitorInstances](#) na Referência de AWS CLI Comandos.

JavaScript

SDK para JavaScript (v3)

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

```
import { EC2Client, UnmonitorInstancesCommand } from "@aws-sdk/client-ec2";
```

```

import { fileURLToPath } from "node:url";
import { parseArgs } from "node:util";

/**
 * Turn off detailed monitoring for the selected instance.
 * @param {{ instanceIds: string[] }} options
 */
export const main = async ({ instanceIds }) => {
  const client = new EC2Client({});
  const command = new UnmonitorInstancesCommand({
    InstanceIds: instanceIds,
  });

  try {
    const { InstanceMonitorings } = await client.send(command);
    const instanceMonitoringsList = InstanceMonitorings.map(
      (im) =>
        ` • Detailed monitoring state for ${im.InstanceId} is
${im.Monitoring.State}.`,
    );
    console.log("Monitoring status:");
    console.log(instanceMonitoringsList.join("\n"));
  } catch (caught) {
    if (
      caught instanceof Error &&
      caught.name === "InvalidInstanceID.NotFound"
    ) {
      console.warn(`${caught.message}`);
    } else {
      throw caught;
    }
  }
};

```

- Para API obter detalhes, consulte [UnmonitorInstances](#) em AWS SDK for JavaScript API Referência.

PowerShell

Ferramentas para PowerShell

Exemplo 1: Este exemplo desativa o monitoramento detalhado da instância especificada.

```
Stop-EC2InstanceMonitoring -InstanceId i-12345678
```

Saída:

```
InstanceId      Monitoring
-----
i-12345678     Amazon.EC2.Model.Monitoring
```

- Para API obter detalhes, consulte [UnmonitorInstances](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Use `UpdateSecurityGroupRuleDescriptionsIngress` com um CLI

Os exemplos de código a seguir mostram como usar o `UpdateSecurityGroupRuleDescriptionsIngress`.

CLI

AWS CLI

Exemplo 1: Para atualizar a descrição de uma regra de grupo de segurança de entrada com uma fonte CIDR

O `update-security-group-rule-descriptions-ingress` exemplo a seguir atualiza a descrição da regra do grupo de segurança para a porta e o intervalo de IPv4 endereços especificados. A descrição 'SSH access from ABC office' substitui qualquer descrição existente para a regra.

```
aws ec2 update-security-group-rule-descriptions-ingress \
  --group-id sg-02f0d35a850ba727f \
  --ip-permissions
  IpProtocol=tcp,FromPort=22,ToPort=22,IpRanges='[{"CidrIp=203.0.113.0/16,Description="SSH
  access from corpnet"}]'
```

Saída:

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

Para obter mais informações, consulte [Regras de grupos de segurança](#) no Guia EC2 do usuário da Amazon.

Exemplo 2: Para atualizar a descrição de uma regra de grupo de segurança de entrada com uma fonte de lista de prefixos

O `update-security-group-rule-descriptions-ingress` exemplo a seguir atualiza a descrição da regra do grupo de segurança para a porta especificada e a lista de prefixos. A descrição 'SSH access from ABC office' substitui qualquer descrição existente para a regra.

```
aws ec2 update-security-group-rule-descriptions-ingress \
  --group-id sg-02f0d35a850ba727f \
  --ip-permissions
  IpProtocol=tcp,FromPort=22,ToPort=22,PrefixListIds='[{"PrefixListId=pl-12345678,Description=SSH access from corpnet"}]'
```

Saída:

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

Para obter mais informações, consulte [Regras de grupos de segurança](#) no Guia EC2 do usuário da Amazon.

- Para API obter detalhes, consulte [UpdateSecurityGroupRuleDescriptionsIngress](#) na Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell

Exemplo 1: atualiza a descrição de uma regra de grupo de segurança de entrada (entrada) existente.


```
$existingInboundRule = Get-EC2SecurityGroupRule -SecurityGroupId
"sgr-1234567890"
$ruleWithUpdatedDescription = [Amazon.EC2.Model.SecurityGroupRuleDescription]@{
    "SecurityGroupId" = $existingInboundRule.SecurityGroupId
    "Description" = "Updated rule description"
}

Update-EC2SecurityGroupRuleIngressDescription -GroupId
$existingInboundRule.GroupId -SecurityGroupRuleDescription
$ruleWithUpdatedDescription
```

Exemplo 2: remove a descrição de uma regra de grupo de segurança de entrada (entrada) existente (omitindo o parâmetro na solicitação).

```
$existingInboundRule = Get-EC2SecurityGroupRule -SecurityGroupId
"sgr-1234567890"
$ruleWithoutDescription = [Amazon.EC2.Model.SecurityGroupRuleDescription]@{
    "SecurityGroupId" = $existingInboundRule.SecurityGroupId
}

Update-EC2SecurityGroupRuleIngressDescription -GroupId
$existingInboundRule.GroupId -SecurityGroupRuleDescription
$ruleWithoutDescription
```

- Para API obter detalhes, consulte [UpdateSecurityGroupRuleDescriptionsIngress](#) em Referência de AWS Tools for PowerShell cmdlet.

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Cenários para o EC2 uso da Amazon AWS SDKs

Os exemplos de código a seguir mostram como implementar cenários comuns na Amazon EC2 com AWS SDKs. Esses cenários mostram como realizar tarefas específicas chamando várias funções na Amazon EC2 ou combinadas com outros Serviços da AWS. Cada cenário inclui um link para o código-fonte completo, em que você pode encontrar instruções sobre como configurar e executar o código.

Os cenários têm como alvo um nível intermediário de experiência para ajudá-lo a entender as ações de serviço no contexto.

Exemplos

- [Crie e gerencie um serviço resiliente usando um AWS SDK](#)

Crie e gerencie um serviço resiliente usando um AWS SDK

Os exemplos de código a seguir mostram como criar um serviço da Web com carga balanceada que retorna recomendações de livros, filmes e músicas. O exemplo mostra como o serviço responde a falhas e como é possível reestruturá-lo para gerar mais resiliência em caso de falhas.

- Use um grupo do Amazon EC2 Auto Scaling para criar instâncias do Amazon Elastic Compute Cloud (AmazonEC2) com base em um modelo de lançamento e para manter o número de instâncias em um intervalo especificado.
- Gerencie e distribua HTTP solicitações com o Elastic Load Balancing.
- Monitore a integridade das instâncias em um grupo do Auto Scaling e encaminhe solicitações somente para instâncias íntegras.
- Execute um servidor web Python em cada EC2 instância para lidar com HTTP as solicitações. O servidor Web responde com recomendações e verificações de integridade.
- Simule um serviço de recomendação com uma tabela do Amazon DynamoDB.
- Controle a resposta do servidor web às solicitações e verificações de saúde atualizando AWS Systems Manager os parâmetros.

.NET

AWS SDK for .NET

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

Execute o cenário interativo em um prompt de comando.

```
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.TargetGroup
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($"http://{endPoint}\n");
}
else
{
    Console.WriteLine(
        "\nCouldn't get a successful response from the load balancer
endpoint. Troubleshoot by\n"
        + "manually verifying that your VPC and security group are
configured correctly and that\n"
        + "you can successfully make a GET request to the load balancer
endpoint:\n");
    Console.WriteLine($"http://{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($"Now, 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($"Even 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($"When 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;
    }

```

Crie uma classe que envolva as ações do Auto Scaling e da AmazonEC2.

```

/// <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 ILogger<AutoScalerWrapper> _logger;

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,
    ILogger<AutoScalerWrapper> logger)
{
    _amazonAutoScaling = amazonAutoScaling;
    _amazonEc2 = amazonEc2;
    _amazonSsm = amazonSsm;
    _amazonIam = amazonIam;
}
```

```

    _logger = logger;

    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)
{
    try
    {
        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;
}
catch (AmazonEC2Exception ec2Exception)
{
    if (ec2Exception.ErrorCode ==
        "InvalidLaunchTemplateName.AlreadyExistsException")
    {
        _logger.LogError($"Could not create the template, the name
        {_launchTemplateName} already exists. " +
            $"Please try again with a unique name.");
    }

    throw;
}
catch (Exception ex)
{
    _logger.LogError($"An error occurred while creating the template.:
    {ex.Message}");
    throw;
}
}
```

```
    }

    /// <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()
    {
        try
        {
            var zoneResponse = await _amazonEc2.DescribeAvailabilityZonesAsync(
                new DescribeAvailabilityZonesRequest());
            return zoneResponse.AvailabilityZones.Select(z =>
z.ZoneName).ToList();
        }
        catch (AmazonEC2Exception ec2Exception)
        {
            _logger.LogError($"An Amazon EC2 error occurred while listing
availability zones.: {ec2Exception.Message}");
            throw;
        }
        catch (Exception ex)
        {
            _logger.LogError($"An error occurred while listing availability
zones.: {ex.Message}");
            throw;
        }
    }
}

/// <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()
{
    try
    {
        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];
    }
    catch (AmazonEC2Exception ec2Exception)
    {

```

```
        if (ec2Exception.ErrorCode == "UnauthorizedOperation")
        {
            _logger.LogError(ec2Exception, $"You do not have the necessary
permissions to describe VPCs.");
        }

        throw;
    }
    catch (Exception ex)
    {
        _logger.LogError(ex, $"An error occurred while describing the vpcs.:
{ex.Message}");
        throw;
    }
}

/// <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)
{
    try
    {
        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;
    }
    catch (AmazonEC2Exception ec2Exception)
    {
        if (ec2Exception.ErrorCode == "InvalidVpcID.NotFound")
        {
            _logger.LogError(ec2Exception, $"The specified VPC ID {vpcId}
does not exist.");
        }

        throw;
    }
    catch (Exception ex)
    {
        _logger.LogError(ex, $"An error occurred while describing the
subnets.: {ex.Message}");
        throw;
    }
}

/// <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 (AmazonEC2Exception ec2Exception)
    {
        if (ec2Exception.ErrorCode ==
"InvalidLaunchTemplateName.NotFoundException")
        {
            _logger.LogError(
                $"Could not delete the template, the name
{_launchTemplateName} was not found.");
        }
    }
}
```

```
    }

    throw;
}
catch (Exception ex)
{
    _logger.LogError($"An error occurred while deleting the template.:
{ex.Message}");
    throw;
}
}

/// <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)
{
    try
    {
        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];
    }
    catch (AmazonEC2Exception ec2Exception)
    {
        if (ec2Exception.ErrorCode == "InvalidInstanceID.NotFound")
        {
            _logger.LogError(ec2Exception, $"Instance {instanceId} not
found");
        }

        throw;
    }
    catch (Exception ex)
    {
        _logger.LogError(ex, $"An error occurred while creating the
template.: {ex.Message}");
        throw;
    }
}

/// <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)
    {
        try
        {
            await _amazonEc2.ReplaceIamInstanceProfileAssociationAsync(
                new ReplaceIamInstanceProfileAssociationRequest()
                {
                    AssociationId = associationId,
                    IamInstanceProfile = new IamInstanceProfileSpecification()
                    {
                        Name = credsProfileName
                    }
                });
            // Allow time before resetting.
            Thread.Sleep(25000);

            await _amazonEc2.RebootInstancesAsync(
                new RebootInstancesRequest(new List<string>() { instanceId }));
            Thread.Sleep(25000);
            var instanceReady = false;
            var retries = 5;
            while (retries-- > 0 && !instanceReady)
            {
                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("Waiting for instance to be running.");
            await WaitForInstanceState(instanceId, InstanceStateName.Running);
            Console.WriteLine("Instance ready.");
        }
    }
}
```

```

        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}");
    }
    catch (AmazonEC2Exception ec2Exception)
    {
        if (ec2Exception.ErrorCode == "InvalidInstanceID.NotFound")
        {
            _logger.LogError(ec2Exception, $"Instance {instanceId} not
found");
        }

        throw;
    }
    catch (Exception ex)
    {
        _logger.LogError(ex, $"An error occurred while replacing the
template.: {ex.Message}");
        throw;
    }
}

/// <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 }
            });
    }

    /// <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 in the specified state.
    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.WriteLine(". ");
    } while (!hasState);

    return hasState;
}
}
```

Crie uma classe que envolva ações do Elastic Load Balancing.

```
/// <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);
        }
    }
}
}
```

Crie uma classe que use o DynamoDB para simular um serviço de recomendação.

```
/// <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.WriteLine($"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.WriteLine("\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.WriteLine(".");
    }
    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");
    }
}
}
```

Crie uma classe que envolva ações do Systems Manager.

```
/// <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 });
    }
}

```

- Para API obter detalhes, consulte os tópicos a seguir em AWS SDK for .NET APIReferência.
 - [AttachLoadBalancerTargetGroups](#)
 - [CreateAutoScalingGroup](#)

- [CreateInstanceProfile](#)
- [CreateLaunchTemplate](#)
- [CreateListener](#)
- [CreateLoadBalancer](#)
- [CreateTargetGroup](#)
- [DeleteAutoScalingGroup](#)
- [DeleteInstanceProfile](#)
- [DeleteLaunchTemplate](#)
- [DeleteLoadBalancer](#)
- [DeleteTargetGroup](#)
- [DescribeAutoScalingGroups](#)
- [DescribeAvailabilityZones](#)
- [DescribeIamInstanceProfileAssociations](#)
- [DescribeInstances](#)
- [DescribeLoadBalancers](#)
- [DescribeSubnets](#)
- [DescribeTargetGroups](#)
- [DescribeTargetHealth](#)
- [DescribeVpcs](#)
- [RebootInstances](#)
- [ReplacelamInstanceProfileAssociation](#)
- [TerminateInstanceInAutoScalingGroup](#)
- [UpdateAutoScalingGroup](#)

Java

SDK para Java 2.x

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

Execute o cenário interativo em um prompt de comando.

```
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 (wasSuccessful) {
        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("""
health check to update
                Note that it can take a minute or two for the
                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);
}
}
```

Crie uma classe que envolva as ações do Auto Scaling e da AmazonEC2.

```
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.getInstanceInformationList();
    for (InstanceInformation info : instanceInformationList) {
        if (info.getInstanceId().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.getInstanceProfile().getInstanceProfileName();
            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.");
    }
}

```

Crie uma classe que envolva ações do Elastic Load Balancing.

```

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 "";
}
}
```

Crie uma classe que use o DynamoDB para simular um serviço de recomendação.

```
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);  
  }  
}
```

Crie uma classe que envolva ações do Systems Manager.

```
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);
    }
}
```

- Para API obter detalhes, consulte os tópicos a seguir em AWS SDK for Java 2.x APIReferência.
 - [AttachLoadBalancerTargetGroups](#)
 - [CreateAutoScalingGroup](#)
 - [CreateInstanceProfile](#)
 - [CreateLaunchTemplate](#)
 - [CreateListener](#)

- [CreateLoadBalancer](#)
- [CreateTargetGroup](#)
- [DeleteAutoScalingGroup](#)
- [DeleteInstanceProfile](#)
- [DeleteLaunchTemplate](#)
- [DeleteLoadBalancer](#)
- [DeleteTargetGroup](#)
- [DescribeAutoScalingGroups](#)
- [DescribeAvailabilityZones](#)
- [DescribelamInstanceProfileAssociations](#)
- [DescribeInstances](#)
- [DescribeLoadBalancers](#)
- [DescribeSubnets](#)
- [DescribeTargetGroups](#)
- [DescribeTargetHealth](#)
- [DescribeVpcs](#)
- [RebootInstances](#)
- [ReplacelamInstanceProfileAssociation](#)
- [TerminateInstanceInAutoScalingGroup](#)
- [UpdateAutoScalingGroup](#)

JavaScript

SDK para JavaScript (v3)

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

Execute o cenário interativo em um prompt de comando.

```
#!/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 "node:url";
```

```
if (process.argv[1] === fileURLToPath(import.meta.url)) {
  parseScenarioArgs(scenarios, {
    name: "Resilient Workflow",
    synopsis:
      "node index.js --scenario <deploy | demo | destroy> [-h|--help] [-y|--yes]
[-v|--verbose]",
    description: "Deploy and interact with scalable EC2 instances.",
  });
}
```

Criar etapas para implantar todos os recursos.

```
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 { saveState } from "@aws-doc-sdk-examples/lib/scenario/steps-common.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 () => {
    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,
    },
  })),
);
}),
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) => {
  const client = new EC2Client({});
  const { Vpcs } = await client.send(
    new DescribeVpcsCommand({
      Filters: [{ Name: "is-default", Values: ["true"] }],
    }),
  );
  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) => {
  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"] },
      ],
    }),
  );
  state.subnets = Subnets.map((subnet) => subnet.SubnetId);
}),
new ScenarioOutput(
  "gotSubnets",
```

```
/**
 * @param {{ subnets: string[] }} state
 */
(state) =>
  MESSAGES.getSubnets.replace("${SUBNETS}", state.subnets.join(", ")),
),
new ScenarioOutput(
  "creatingLoadBalancerTargetGroup",
  MESSAGES.creatingLoadBalancerTargetGroup.replace(
    "${TARGET_GROUP_NAME}",
    NAMES.loadBalancerTargetGroupName,
  ),
),
new ScenarioAction("createLoadBalancerTargetGroup", async (state) => {
  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,
    })),
  );
  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) => {
  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] },
  );
}),
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) => {
  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 },
      ],
    }),
  );
  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) => {
  const client = new AutoScalingClient({});
  await client.send(
    new AttachLoadBalancerTargetGroupsCommand({
      AutoScalingGroupName: NAMES.autoScalingGroupName,
      TargetGroupARNs: [state.targetGroupArn],
    }),
  );
}),
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),
      );
    }
    return MESSAGES.noIpRules;
  },
),
new ScenarioInput(
  "shouldAddInboundRule",
  /**
   * @param {{ myIpRules: any[] }} state
   */
  (state) => {
    if (state.myIpRules.length > 0) {
      return false;
    }
    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);
  }
  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,
    );
  }
}),
saveState,
];
```

Criar etapas para executar a demonstração.

```
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) => {
  const client = new ElasticLoadBalancingV2Client({});
  const { TargetGroups } = await client.send(
    new DescribeTargetGroupsCommand({
      Names: [NAMES.loadBalancerTargetGroupName],
    }),
  );

  const { TargetHealthDescriptions } = await client.send(
    new DescribeTargetHealthCommand({
      TargetGroupArn: TargetGroups[0].TargetGroupArn,
    }),
  );
  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];
    const ec2Client = new EC2Client({});
    const { IamInstanceProfileAssociations } = await ec2Client.send(
      new DescribeIamInstanceProfileAssociationsCommand({
        Filters: [
          { Name: "instance-id", Values: [state.targetInstance.InstanceId] },
        ],
      }),
    );
    state.instanceProfileAssociationId =
      IamInstanceProfileAssociations[0].AssociationId;
    await retry({ intervalInMs: 1000, maxRetries: 30 }, () =>
      ec2Client.send(
        new ReplaceIamInstanceProfileAssociationCommand({

```

```
        AssociationId: state.instanceProfileAssociationId,
        IamInstanceProfile: { Name: NAMES.ssmOnlyInstanceProfileName },
    })),
    ),
);

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(
```

```

        "${INSTANCE_ID}",
        state.targetInstance.InstanceId,
    ),
),
loadBalancerLoop,
new ScenarioInput(
    "deepHealthCheckConfirmation",
    MESSAGES.demoDeepHealthCheckConfirmation,
    { type: "confirm" },
),
new ScenarioAction("deepHealthCheckExit", (state) => {
    if (!state.deepHealthCheckConfirmation) {
        process.exit();
    }
}),
new ScenarioAction("deepHealthCheck", async () => {
    const client = new SSMClient({});
    await client.send(
        new PutParameterCommand({
            Name: NAMES.ssmHealthCheckKey,
            Value: "deep",
            Overwrite: true,
            Type: "String",
        }),
    );
}),
new ScenarioOutput("testDeepHealthCheck", MESSAGES.demoTestDeepHealthCheck),
loadBalancerLoop,
new ScenarioInput(
    "killInstanceConfirmation",
    /**
     * @param {{ targetInstance: import('@aws-sdk/client-
    ssm').InstanceInformation }} state
     */
    (state) =>
        MESSAGES.demoKillInstanceConfirmation.replace(
            "${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",
    }),
);
const { InstanceProfile } = await iamClient.send(
    new CreateInstanceProfileCommand({
        InstanceProfileName: NAMES.ssmOnlyInstanceProfileName,
    }),
);
await waitUntilInstanceProfileExists(
    { client: iamClient },
    { InstanceProfileName: NAMES.ssmOnlyInstanceProfileName },
);
await iamClient.send(
    new AddRoleToInstanceProfileCommand({
        InstanceProfileName: NAMES.ssmOnlyInstanceProfileName,
        RoleName: NAMES.ssmOnlyRoleName,
    }),
);

return InstanceProfile;
}

```

Criar etapas para destruir todos os recursos.

```
import { unlinkSync } from "node:fs";
```

```
import { DynamoDBClient, DeleteTableCommand } from "@aws-sdk/client-dynamodb";
import {
  EC2Client,
  DeleteKeyPairCommand,
  DeleteLaunchTemplateCommand,
  RevokeSecurityGroupIngressCommand,
} 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 { loadState } from "@aws-doc-sdk-examples/lib/scenario/steps-common.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 = [
  loadState,
  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(
        "${TABLE_NAME}",
        NAMES.tableName,
      );
    }
    return MESSAGES.deletedTable.replace("${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,
      );
    }
  })
];
```

```
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("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName);
    }
    return MESSAGES.detachedPolicyFromRole
        .replace("${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(
      "${INSTANCE_POLICY_NAME}",
      NAMES.instancePolicyName,
    );
  }
  return MESSAGES.deletedPolicy.replace(
    "${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);
  }
  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,
        );
      }
      return MESSAGES.deletedInstanceRole.replace(
        "${INSTANCE_ROLE_NAME}",
        NAMES.instanceRoleName,
      );
    })),
    new ScenarioAction("deleteInstanceProfile", async (state) => {
      try {
        const client = new IAMClient({});
        await client.send(
          new DeleteInstanceProfileCommand({
            InstanceProfileName: NAMES.instanceProfileName,
          }),
        );
      } catch (e) {
        state.deleteInstanceProfileError = e;
      }
    })),
    new ScenarioOutput("deleteInstanceProfileResult", (state) => {
      if (state.deleteInstanceProfileError) {
        console.error(state.deleteInstanceProfileError);
        return MESSAGES.deleteInstanceProfileError.replace(
          "${INSTANCE_PROFILE_NAME}",
```

```
        NAMES.instanceProfileName,
    );
}
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,
        );
    }
    return MESSAGES.deletedAutoScalingGroup.replace(
        "${AUTO_SCALING_GROUP_NAME}",
        NAMES.autoScalingGroupName,
    );
}),
new ScenarioAction("deleteLaunchTemplate", async (state) => {
    const client = new EC2Client({});
    try {
        await client.send(
            new DeleteLaunchTemplateCommand({
                LaunchTemplateName: NAMES.launchTemplateName,
            }),
        );
    } catch (e) {
        state.deleteLaunchTemplateError = e;
    }
}),
new ScenarioOutput("deleteLaunchTemplateResult", (state) => {
```

```
    if (state.deleteLaunchTemplateError) {
      console.error(state.deleteLaunchTemplateError);
      return MESSAGES.deleteLaunchTemplateError.replace(
        "${LAUNCH_TEMPLATE_NAME}",
        NAMES.launchTemplateName,
      );
    }
    return MESSAGES.deletedLaunchTemplate.replace(
      "${LAUNCH_TEMPLATE_NAME}",
      NAMES.launchTemplateName,
    );
  })),
  new ScenarioAction("deleteLoadBalancer", async (state) => {
    try {
      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.");
        }
      });
    } catch (e) {
      state.deleteLoadBalancerError = e;
    }
  })),
  new ScenarioOutput("deleteLoadBalancerResult", (state) => {
    if (state.deleteLoadBalancerError) {
      console.error(state.deleteLoadBalancerError);
      return MESSAGES.deleteLoadBalancerError.replace(
        "${LB_NAME}",
        NAMES.loadBalancerName,
      );
    }
    return MESSAGES.deletedLoadBalancer.replace(
      "${LB_NAME}",
      NAMES.loadBalancerName,
    );
  })),
```

```
new ScenarioAction("deleteLoadBalancerTargetGroup", async (state) => {
  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;
  }
}),
new ScenarioOutput("deleteLoadBalancerTargetGroupResult", (state) => {
  if (state.deleteLoadBalancerTargetGroupError) {
    console.error(state.deleteLoadBalancerTargetGroupError);
    return MESSAGES.deleteLoadBalancerTargetGroupError.replace(
      "${TARGET_GROUP_NAME}",
      NAMES.loadBalancerTargetGroupName,
    );
  }
  return MESSAGES.deletedLoadBalancerTargetGroup.replace(
    "${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);
    }
    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);
    }
    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");
    }
    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(
            "${INSTANCE_PROFILE_NAME}",
            NAMES.ssmOnlyInstanceProfileName,
        );
    }
    return MESSAGES.deletedSsmOnlyInstanceProfile.replace(
        "${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(
        "${POLICY_NAME}",
        NAMES.ssmOnlyPolicyName,
      );
    }
    return MESSAGES.deletedSsmOnlyPolicy.replace(
      "${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,
      );
    }
  }));
```

```

    }
    return MESSAGES.deletedSsmOnlyRole.replace(
        "${ROLE_NAME}",
        NAMES.ssmOnlyRoleName,
    );
}),
new ScenarioAction(
    "revokeSecurityGroupIngress",
    async (
        /** @type {{ myIp: string, defaultSecurityGroup: { GroupId: string } }} */
state,
    ) => {
        const ec2Client = new EC2Client({});

        try {
            await ec2Client.send(
                new RevokeSecurityGroupIngressCommand({
                    GroupId: state.defaultSecurityGroup.GroupId,
                    CidrIp: `${state.myIp}/32`,
                    FromPort: 80,
                    ToPort: 80,
                    IpProtocol: "tcp",
                }),
            );
        } catch (e) {
            state.revokeSecurityGroupIngressError = e;
        }
    },
),
new ScenarioOutput("revokeSecurityGroupIngressResult", (state) => {
    if (state.revokeSecurityGroupIngressError) {
        console.error(state.revokeSecurityGroupIngressError);
        return MESSAGES.revokeSecurityGroupIngressError.replace(
            "${IP}",
            state.myIp,
        );
    }
    return MESSAGES.revokedSecurityGroupIngress.replace("${IP}", state.myIp);
}),
];

/**
 * @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;
    }
    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.`);
}
```

- Para API obter detalhes, consulte os tópicos a seguir em AWS SDK for JavaScript APIReferência.
 - [AttachLoadBalancerTargetGroups](#)
 - [CreateAutoScalingGroup](#)
 - [CreateInstanceProfile](#)
 - [CreateLaunchTemplate](#)
 - [CreateListener](#)
 - [CreateLoadBalancer](#)
 - [CreateTargetGroup](#)
 - [DeleteAutoScalingGroup](#)
 - [DeleteInstanceProfile](#)
 - [DeleteLaunchTemplate](#)
 - [DeleteLoadBalancer](#)

- [DeleteTargetGroup](#)
- [DescribeAutoScalingGroups](#)
- [DescribeAvailabilityZones](#)
- [DescribeIamInstanceProfileAssociations](#)
- [DescribeInstances](#)
- [DescribeLoadBalancers](#)
- [DescribeSubnets](#)
- [DescribeTargetGroups](#)
- [DescribeTargetHealth](#)
- [DescribeVpcs](#)
- [RebootInstances](#)
- [ReplacelamInstanceProfileAssociation](#)
- [TerminateInstanceInAutoScalingGroup](#)
- [UpdateAutoScalingGroup](#)

Python

SDK para Python (Boto3)

Note

Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no [Repositório de exemplos de código da AWS](#).

Execute o cenário interativo em um prompt de comando.

```
class Runner:
    """
    Manages the deployment, demonstration, and destruction of resources for the
    resilient service.
    """

    def __init__(
        self,
        resource_path: str,
```

```

        recommendation: RecommendationService,
        autoscaler: AutoScalingWrapper,
        loadbalancer: ElasticLoadBalancerWrapper,
        param_helper: ParameterHelper,
    ):
        """
        Initializes the Runner class with the necessary parameters.

        :param resource_path: The path to resource files used by this example,
        such as IAM policies and instance scripts.
        :param recommendation: An instance of the RecommendationService class.
        :param autoscaler: An instance of the AutoScaler class.
        :param loadbalancer: An instance of the LoadBalancer class.
        :param param_helper: An instance of the ParameterHelper class.
        """
        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

        prefix = "doc-example-resilience"
        self.target_group_name = f"{prefix}-tg"
        self.load_balancer_name = f"{prefix}-lb"

    def deploy(self) -> None:
        """
        Deploys the resources required for the resilient service, including the
        DynamoDB table,
        EC2 instances, Auto Scaling group, and load balancer.
        """
        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"

        logging.info("Starting deployment of resources for the resilient
        service.")

        logging.info(
            "Creating and populating DynamoDB table '%s'.",
            self.recommendation.table_name,

```

```
)
self.recommendation.create()
self.recommendation.populate(recommendations_path)

logging.info(
    "Creating an EC2 launch template with the startup script '%s'.",
    startup_script,
)
self.autoscaler.create_template(startup_script, instance_policy)

logging.info(
    "Creating an EC2 Auto Scaling group across multiple Availability
Zones."
)
zones = self.autoscaler.create_autoscaling_group(3)

logging.info("Creating variables that control the flow of the demo.")
self.param_helper.reset()

logging.info("Creating Elastic Load Balancing target group and load
balancer.")

vpc = self.autoscaler.get_default_vpc()
subnets = self.autoscaler.get_subnets(vpc["VpcId"], zones)
target_group = self.loadbalancer.create_target_group(
    self.target_group_name, self.protocol, self.port, vpc["VpcId"]
)
self.loadbalancer.create_load_balancer(
    self.load_balancer_name, [subnet["SubnetId"] for subnet in subnets]
)
self.loadbalancer.create_listener(self.load_balancer_name, target_group)

self.autoscaler.attach_load_balancer_target_group(target_group)

logging.info("Verifying access to the load balancer endpoint.")
endpoint = self.loadbalancer.get_endpoint(self.load_balancer_name)
lb_success = self.loadbalancer.verify_load_balancer_endpoint(endpoint)
current_ip_address = requests.get("http://
checkip.amazonaws.com").text.strip()

if not lb_success:
    logging.warning(
        "Couldn't connect to the load balancer. Verifying that the port
is open..."
    )
```



```
    )
    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:
        logging.warning(
            "The default security group for your VPC must allow access
from this computer."
        )
        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(endpoint)

    if lb_success:
        logging.info(
            "Load balancer is ready. Access it at: http://%s",
current_ip_address
        )
    else:
        logging.error(
```

```
        "Couldn't get a successful response from the load balancer
        endpoint. Please verify your VPC and security group settings."
    )

    def demo_choices(self) -> None:
        """
        Presents choices for interacting with the deployed service, such as
        sending requests to
        the load balancer or checking the health of the targets.
        """
        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:
            logging.info("Choose an action to interact with the service.")
            choice = q.choose("Which action would you like to take? ", actions)
            if choice == 0:
                logging.info("Sending a GET request to the load balancer
                endpoint.")
                endpoint =
                self.loadbalancer.get_endpoint(self.load_balancer_name)
                logging.info("GET http://%s", endpoint)
                response = requests.get(f"http://{endpoint}")
                logging.info("Response: %s", response.status_code)
                if response.headers.get("content-type") == "application/json":
                    pp(response.json())
            elif choice == 1:
                logging.info("Checking the health of load balancer targets.")
                health =
                self.loadbalancer.check_target_health(self.target_group_name)
                for target in health:
                    state = target["TargetHealth"]["State"]
                    logging.info(
                        "Target %s on port %d is %s",
                        target["Target"]["Id"],
                        target["Target"]["Port"],
                        state,
                    )
                if state != "healthy":
                    logging.warning(
                        "%s: %s",
```

```
                target["TargetHealth"]["Reason"],
                target["TargetHealth"]["Description"],
            )
            logging.info(
                "Note that it can take a minute or two for the health check
to update."
            )
        elif choice == 2:
            logging.info("Proceeding to the next part of the demo.")

def demo(self) -> None:
    """
    Runs the demonstration, showing how the service responds to different
failure scenarios
and how a resilient architecture can keep the service running.
    """
    ssm_only_policy = f"{self.resource_path}/ssm_only_policy.json"

    logging.info("Resetting parameters to starting values for the demo.")
    self.param_helper.reset()

    logging.info(
        "Starting demonstration of the service's resilience under various
failure conditions."
    )
    self.demo_choices()

    logging.info(
        "Simulating failure by changing the Systems Manager parameter to a
non-existent table."
    )
    self.param_helper.put(self.param_helper.table, "this-is-not-a-table")
    logging.info("Sending GET requests will now return failure codes.")
    self.demo_choices()

    logging.info("Switching to static response mode to mitigate failure.")
    self.param_helper.put(self.param_helper.failure_response, "static")
    logging.info("Sending GET requests will now return static responses.")
    self.demo_choices()

    logging.info("Restoring normal operation of the recommendation service.")
    self.param_helper.put(self.param_helper.table,
self.recommendation.table_name)
```

```
        logging.info(
            "Introducing a failure by assigning bad credentials to one of the
instances."
        )
        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)
        logging.info(
            "Replacing instance profile with bad credentials for instance %s.",
            bad_instance_id,
        )
        self.autoscaler.replace_instance_profile(
            bad_instance_id,
            self.autoscaler.bad_creds_profile_name,
            instance_profile["AssociationId"],
        )
        logging.info(
            "Sending GET requests may return either a valid recommendation or a
static response."
        )
        self.demo_choices()

        logging.info("Implementing deep health checks to detect unhealthy
instances.")
        self.param_helper.put(self.param_helper.health_check, "deep")
        logging.info("Checking the health of the load balancer targets.")
        self.demo_choices()

        logging.info(
            "Terminating the unhealthy instance to let the auto scaler replace
it."
        )
        self.autoscaler.terminate_instance(bad_instance_id)
        logging.info("The service remains resilient during instance
replacement.")
        self.demo_choices()
```

```
        logging.info("Simulating a complete failure of the recommendation
service.")
        self.param_helper.put(self.param_helper.table, "this-is-not-a-table")
        logging.info(
            "All instances will report as unhealthy, but the service will still
return static responses."
        )
        self.demo_choices()
        self.param_helper.reset()

    def destroy(self, automation=False) -> None:
        """
        Destroys all resources created for the demo, including the load balancer,
Auto Scaling group,
        EC2 instances, and DynamoDB table.
        """
        logging.info(
            "This concludes the demo. Preparing to clean up all AWS resources
created during the demo."
        )
        if automation:
            cleanup = True
        else:
            cleanup = q.ask(
                "Do you want to clean up all demo resources? (y/n) ", q.is_yesno
            )

        if cleanup:
            logging.info("Deleting load balancer and related resources.")
            self.loadbalancer.delete_load_balancer(self.load_balancer_name)
            self.loadbalancer.delete_target_group(self.target_group_name)
            self.autoscaler.delete_autoscaling_group(self.autoscaler.group_name)
            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,
            )
            logging.info("Deleting DynamoDB table and other resources.")
            self.recommendation.destroy()
        else:
            logging.warning(
                "Resources have not been deleted. Ensure you clean them up
manually to avoid unexpected charges."
```

```
)

def main() -> None:
    """
    Main function to parse arguments and run the appropriate actions for the
    demo.
    """
    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()

    logging.info("Starting the Resilient Service demo.")

    prefix = "doc-example-resilience"

    # Service Clients
    ddb_client = boto3.client("dynamodb")
    elb_client = boto3.client("elbv2")
    autoscaling_client = boto3.client("autoscaling")
    ec2_client = boto3.client("ec2")
    ssm_client = boto3.client("ssm")
    iam_client = boto3.client("iam")

    # Wrapper instantiations
    recommendation = RecommendationService(
        "doc-example-recommendation-service", ddb_client
    )
    autoscaling_wrapper = AutoScalingWrapper(
        prefix,
```

```

        "t3.micro",
        "/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2",
        autoscaling_client,
        ec2_client,
        ssm_client,
        iam_client,
    )
    elb_wrapper = ElasticLoadBalancerWrapper(elb_client)
    param_helper = ParameterHelper(recommendation.table_name, ssm_client)

    # Demo invocation
    runner = Runner(
        args.resource_path,
        recommendation,
        autoscaling_wrapper,
        elb_wrapper,
        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()

    logging.info("Demo completed successfully.")

if __name__ == "__main__":
    logging.basicConfig(level=logging.INFO, format="%(levelname)s: %(message)s")
    main()

```

Crie uma classe que envolva as ações do Auto Scaling e da AmazonEC2.

```

class AutoScalingWrapper:
    """
    Encapsulates Amazon EC2 Auto Scaling and EC2 management actions.
    """

```

```
def __init__(
    self,
    resource_prefix: str,
    inst_type: str,
    ami_param: str,
    autoscaling_client: boto3.client,
    ec2_client: boto3.client,
    ssm_client: boto3.client,
    iam_client: boto3.client,
):
    """
    Initializes the AutoScaler class with the necessary parameters.

    :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
    sts_client = boto3.client("sts")
    self.account_id = sts_client.get_caller_identity()["Account"]

    self.key_pair_name = f"{resource_prefix}-key-pair"
    self.launch_template_name = f"{resource_prefix}-template-"
    self.group_name = f"{resource_prefix}-group"

    # Happy path
    self.instance_policy_name = f"{resource_prefix}-pol"
    self.instance_role_name = f"{resource_prefix}-role"
    self.instance_profile_name = f"{resource_prefix}-prof"

    # Failure mode
    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"

def create_policy(self, policy_file: str, policy_name: str) -> str:
    """
    Creates a new IAM policy or retrieves the ARN of an existing policy.

    :param policy_file: The path to a JSON file that contains the policy
    definition.
    :param policy_name: The name to give the created policy.
    :return: The ARN of the created or existing policy.
    """
    with open(policy_file) as file:
        policy_doc = file.read()

    try:
        response = self.iam_client.create_policy(
            PolicyName=policy_name, PolicyDocument=policy_doc
        )
        policy_arn = response["Policy"]["Arn"]
        log.info(f"Policy '{policy_name}' created successfully. ARN:
{policy_arn}")
        return policy_arn

    except ClientError as err:
        if err.response["Error"]["Code"] == "EntityAlreadyExists":
            # If the policy already exists, get its ARN
            response = self.iam_client.get_policy(
                PolicyArn=f"arn:aws:iam::{self.account_id}:policy/
{policy_name}"
            )
            policy_arn = response["Policy"]["Arn"]
            log.info(f"Policy '{policy_name}' already exists. ARN:
{policy_arn}")
            return policy_arn
        log.error(f"Full error:\n\t{err}")

def create_role(self, role_name: str, assume_role_doc: dict) -> str:
    """
    Creates a new IAM role or retrieves the ARN of an existing role.

    :param role_name: The name to give the created role.
    :param assume_role_doc: The assume role policy document that specifies
    which
```

```

        entities can assume the role.
    """
    :return: The ARN of the created or existing role.
    """
    try:
        response = self.iam_client.create_role(
            RoleName=role_name,
            AssumeRolePolicyDocument=json.dumps(assume_role_doc)
        )
        role_arn = response["Role"]["Arn"]
        log.info(f"Role '{role_name}' created successfully. ARN: {role_arn}")
        return role_arn

    except ClientError as err:
        if err.response["Error"]["Code"] == "EntityAlreadyExists":
            # If the role already exists, get its ARN
            response = self.iam_client.get_role(RoleName=role_name)
            role_arn = response["Role"]["Arn"]
            log.info(f"Role '{role_name}' already exists. ARN: {role_arn}")
            return role_arn
        log.error(f"Full error:\n\t{err}")

def attach_policy(
    self,
    role_name: str,
    policy_arn: str,
    aws_managed_policies: Tuple[str, ...] = (),
) -> None:
    """
    Attaches an IAM policy to a role and optionally attaches additional AWS-
    managed policies.

    :param role_name: The name of the role to attach the policy to.
    :param policy_arn: The ARN of the policy to attach.
    :param aws_managed_policies: A tuple of AWS-managed policy names to
    attach to the role.
    """
    try:
        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(f"Attached policy {policy_arn} to role {role_name}.")
    except ClientError as err:
        log.error(f"Failed to attach policy {policy_arn} to role
{role_name}.")
        log.error(f"Full error:\n\t{err}")

def create_instance_profile(
    self,
    policy_file: str,
    policy_name: str,
    role_name: str,
    profile_name: str,
    aws_managed_policies: Tuple[str, ...] = (),
) -> str:
    """
    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",
            }
        ]
    }

```

```

        }
    ],
}
policy_arn = self.create_policy(policy_file, policy_name)
self.create_role(role_name, assume_role_doc)
self.attach_policy(role_name, policy_arn, aws_managed_policies)

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(
profile_name
            "Instance profile %s already exists, nothing to do.",
profile_name
        )
        log.error(f"Full error:\n\t{err}")
    return profile_arn

def get_instance_profile(self, instance_id: str) -> Dict[str, Any]:
    """
    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]}]
    )
    if not response["IamInstanceProfileAssociations"]:
        log.info(f"No instance profile found for instance
{instance_id}.")
        profile_data = response["IamInstanceProfileAssociations"][0]
        log.info(f"Retrieved instance profile for instance {instance_id}.")
        return profile_data
    except ClientError as err:
        log.error(
            f"Failed to retrieve instance profile for instance
{instance_id}."
        )
        error_code = err.response["Error"]["Code"]
        if error_code == "InvalidInstanceID.NotFound":
            log.error(f"The instance ID '{instance_id}' does not exist.")
            log.error(f"Full error:\n\t{err}")

def replace_instance_profile(
    self,
    instance_id: str,
    new_instance_profile_name: str,
    profile_association_id: str,
) -> None:
    """
    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 restart.
    :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)

    self.ec2_client.reboot_instances(InstanceIds=[instance_id])
    log.info("Rebooting instance %s.", instance_id)
    waiter = self.ec2_client.get_waiter("instance_running")
    log.info("Waiting for instance %s to be running.", instance_id)
    waiter.wait(InstanceIds=[instance_id])
    log.info("Instance %s is now running.", instance_id)

    self.ssm_client.send_command(
        InstanceIds=[instance_id],
        DocumentName="AWS-RunShellScript",
        Parameters={"commands": ["cd / && sudo python3 server.py 80"]},
    )
    log.info(f"Restarted the Python web server on instance
    '{instance_id}'.")
    except ClientError as err:
        log.error("Failed to replace instance profile.")
        error_code = err.response["Error"]["Code"]
        if error_code == "InvalidAssociationID.NotFound":
            log.error(
                f"Association ID '{profile_association_id}' does not exist."
                "Please check the association ID and try again."
            )
        if error_code == "InvalidInstanceId":
            log.error(
                f"The specified instance ID '{instance_id}' does not exist or
            is not available for SSM. "
                f"Please verify the instance ID and try again."
            )
        log.error(f"Full error:\n\t{err}")

    def delete_instance_profile(self, profile_name: str, role_name: str) -> None:
        """
        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:
    log.error(
        f"Couldn't delete instance profile {profile_name} or detach "
        f"policies and delete role {role_name}: {err}"
    )
    if err.response["Error"]["Code"] == "NoSuchEntity":
        log.info(
            "Instance profile %s doesn't exist, nothing to do.",
profile_name
        )

def create_key_pair(self, key_pair_name: str) -> None:
    """
    Creates a new key pair.

    :param key_pair_name: The name of the key pair to create.
    """
    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:
        error_code = err.response["Error"]["Code"]
        log.error(f"Failed to create key pair {key_pair_name}.")
        if error_code == "InvalidKeyPair.Duplicate":
            log.error(f"A key pair with the name '{key_pair_name}' already
exists.")
        log.error(f"Full error:\n\t{err}")

def delete_key_pair(self) -> None:
    """
    Deletes a key pair.
    """
    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:
        log.error(f"Couldn't delete key pair '{self.key_pair_name}'.")
        log.error(f"Full error:\n\t{err}")
    except FileNotFoundError as err:
        log.info("Key pair %s doesn't exist, nothing to do.",
self.key_pair_name)
        log.error(f"Full error:\n\t{err}")

def create_template(
    self, server_startup_script_file: str, instance_policy_file: str
) -> Dict[str, Any]:
    """
    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:
        # Create key pair and instance profile
        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,
        )

        # Read the startup script
        with open(server_startup_script_file) as file:
            start_server_script = file.read()

        # Get the latest AMI ID
        ami_latest = self.ssm_client.get_parameter(Name=self.ami_param)
        ami_id = ami_latest["Parameter"]["Value"]

        # Create the launch template
        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(
            f"Created launch template {self.launch_template_name} for AMI
{ami_id} on {self.inst_type}."
        )
    
```

```
    except ClientError as err:
        log.error(f"Failed to create launch template
{self.launch_template_name}.")
        error_code = err.response["Error"]["Code"]
        if error_code == "InvalidLaunchTemplateName.AlreadyExistsException":
            log.info(
                f"Launch template {self.launch_template_name} already exists,
nothing to do."
            )
        log.error(f"Full error:\n\t{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,
            )
        log.error(f"Full error:\n\t{err}")

def get_availability_zones(self) -> List[str]:
    """
    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"]]
    log.info(f"Retrieved {len(zones)} availability zones: {zones}.")
except ClientError as err:
    log.error("Failed to retrieve availability zones.")
    log.error(f"Full error:\n\t{err}")
else:
    return zones

def create_autoscaling_group(self, group_size: int) -> List[str]:
    """
    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.
    """
    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(
            f"Created EC2 Auto Scaling group {self.group_name} with
            availability zones {zones}."
        )
    except ClientError as err:
        error_code = err.response["Error"]["Code"]
        if error_code == "AlreadyExists":
            log.info(
                f"EC2 Auto Scaling group {self.group_name} already exists,
                nothing to do."
            )
        else:
```

```
        log.error(f"Failed to create EC2 Auto Scaling group
{self.group_name}.")
        log.error(f"Full error:\n\t{err}")
    else:
        return zones

def get_instances(self) -> List[str]:
    """
    Gets data about the instances in the EC2 Auto Scaling group.

    :return: A list of instance IDs in the Auto Scaling group.
    """
    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"]
        ]
        log.info(
            f"Retrieved {len(instance_ids)} instances for Auto Scaling group
{self.group_name}."
        )
    except ClientError as err:
        error_code = err.response["Error"]["Code"]
        log.error(
            f"Failed to retrieve instances for Auto Scaling group
{self.group_name}."
        )
        if error_code == "ResourceNotFound":
            log.error(f"The Auto Scaling group '{self.group_name}' does not
exist.")
        log.error(f"Full error:\n\t{err}")
    else:
        return instance_ids

def terminate_instance(self, instance_id: str, decrementssetting=False) ->
None:
    """
    Terminates an instance 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.
:param decrementssetting: If True, do not replace terminated instances.
"""
try:
    self.autoscaling_client.terminate_instance_in_auto_scaling_group(
        InstanceId=instance_id,
        ShouldDecrementDesiredCapacity=decrementssetting,
    )
    log.info("Terminated instance %s.", instance_id)

    # Adding a waiter to ensure the instance is terminated
    waiter = self.ec2_client.get_waiter("instance_terminated")
    log.info("Waiting for instance %s to be terminated...", instance_id)
    waiter.wait(InstanceIds=[instance_id])
    log.info(
        f"Instance '{instance_id}' has been terminated and will be
replaced."
    )

except ClientError as err:
    error_code = err.response["Error"]["Code"]
    log.error(f"Failed to terminate instance '{instance_id}'.")
    if error_code == "ScalingActivityInProgressFault":
        log.error(
            "Scaling activity is currently in progress. "
            "Wait for the scaling activity to complete before attempting
to terminate the instance again."
        )
    elif error_code == "ResourceContentionFault":
        log.error(
            "The request failed due to a resource contention issue. "
            "Ensure that no conflicting operations are being performed on
the resource."
        )
    log.error(f"Full error:\n\t{err}")

def attach_load_balancer_target_group(
    self, lb_target_group: Dict[str, Any]
) -> None:
    """
    Attaches an Elastic Load Balancing (ELB) target group to this EC2 Auto
Scaling group.

```

```

    The target group specifies how the load balancer forwards 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:
    error_code = err.response["Error"]["Code"]
    log.error(
        f"Failed to attach load balancer target group
'{lb_target_group['TargetGroupName']}'."
    )
    if error_code == "ResourceContentionFault":
        log.error(
            "The request failed due to a resource contention issue. "
            "Ensure that no conflicting operations are being performed on
the resource."
        )
    elif error_code == "ServiceLinkedRoleFailure":
        log.error(
            "The operation failed because the service-linked role is not
ready or does not exist. "
            "Check that the service-linked role exists and is correctly
configured."
        )
    log.error(f"Full error:\n\t{err}")

def delete_autoscaling_group(self, group_name: str) -> None:
    """
    Terminates all instances in the group, then deletes the EC2 Auto Scaling
group.

```

```

:param group_name: The name of the group to delete.
"""
try:
    response = self.autoscaling_client.describe_auto_scaling_groups(
        AutoScalingGroupNames=[group_name]
    )
    groups = response.get("AutoScalingGroups", [])
    if len(groups) > 0:
        self.autoscaling_client.update_auto_scaling_group(
            AutoScalingGroupName=group_name, MinSize=0
        )
        instance_ids = [inst["InstanceId"] for inst in groups[0]
["Instances"]]
        for inst_id in instance_ids:
            self.terminate_instance(inst_id)

        # Wait for all instances to be terminated
        if instance_ids:
            waiter = self.ec2_client.get_waiter("instance_terminated")
            log.info("Waiting for all instances to be terminated...")
            waiter.wait(InstanceIds=instance_ids)
            log.info("All instances have been terminated.")
        else:
            log.info(f"No groups found named '{group_name}'! Nothing to do.")
except ClientError as err:
    error_code = err.response["Error"]["Code"]
    log.error(f"Failed to delete Auto Scaling group '{group_name}'.")
    if error_code == "ScalingActivityInProgressFault":
        log.error(
            "Scaling activity is currently in progress. "
            "Wait for the scaling activity to complete before attempting
to delete the group again."
        )
    elif error_code == "ResourceContentionFault":
        log.error(
            "The request failed due to a resource contention issue. "
            "Ensure that no conflicting operations are being performed on
the group."
        )
    log.error(f"Full error:\n\t{err}")

def get_default_vpc(self) -> Dict[str, Any]:
    """

```

```

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:
    error_code = err.response["Error"]["Code"]
    log.error("Failed to retrieve the default VPC.")
    if error_code == "UnauthorizedOperation":
        log.error(
            "You do not have the necessary permissions to describe VPCs.

            "Ensure that your AWS IAM user or role has the correct
permissions."
        )
    elif error_code == "InvalidParameterValue":
        log.error(
            "One or more parameters are invalid. Check the request
parameters."
        )

    log.error(f"Full error:\n\t{err}")
else:
    if "Vpcs" in response and response["Vpcs"]:
        log.info(f"Retrieved default VPC: {response['Vpcs'][0]
['VpcId']}")
        return response["Vpcs"][0]
    else:
        pass

def verify_inbound_port(
    self, vpc: Dict[str, Any], port: int, ip_address: str
) -> Tuple[Dict[str, Any], bool]:
    """
    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 specified 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(f"Found default security group {sec_group['GroupId']}.")

        for ip_perm in sec_group["IpPermissions"]:
            if ip_perm.get("FromPort", 0) == port:
                log.info(f"Found inbound rule: {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(
                        f"The inbound rule does not appear to be open to
                        either this computer's IP "
                        f"address of {ip_address}, to all IP addresses
                        (0.0.0.0/0), or to a prefix list ID."
                    )
            else:
                break
    except ClientError as err:
        error_code = err.response["Error"]["Code"]

```

```

        log.error(
            f"Failed to verify inbound rule for port {port} for VPC
{vpc['VpcId']}."
        )
        if error_code == "InvalidVpcID.NotFound":
            log.error(
                f"The specified VPC ID '{vpc['VpcId']}' does not exist.
Please check the VPC ID."
            )
            log.error(f"Full error:\n\t{err}")
        else:
            return sec_group, port_is_open

def open_inbound_port(self, sec_group_id: str, port: int, ip_address: str) ->
None:
    """
    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:
        error_code = err.response["Error"]["Code"]
        log.error(
            f"Failed to authorize ingress to security group '{sec_group_id}'
on port {port} from {ip_address}."

```

```

    )
    if error_code == "InvalidGroupId.Malformed":
        log.error(
            "The security group ID is malformed. "
            "Please verify that the security group ID is correct."
        )
    elif error_code == "InvalidPermission.Duplicate":
        log.error(
            "The specified rule already exists in the security group. "
            "Check the existing rules for this security group."
        )
    log.error(f"Full error:\n\t{err}")

def get_subnets(self, vpc_id: str, zones: List[str] = None) -> List[Dict[str,
Any]]:
    """
    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.
    """
    # Ensure that 'zones' is a list, even if None is passed
    if zones is None:
        zones = []
    try:
        paginator = self.ec2_client.get_paginator("describe_subnets")
        page_iterator = paginator.paginate(
            Filters=[
                {"Name": "vpc-id", "Values": [vpc_id]},
                {"Name": "availability-zone", "Values": zones},
                {"Name": "default-for-az", "Values": ["true"]},
            ]
        )

        subnets = []
        for page in page_iterator:
            subnets.extend(page["Subnets"])

        log.info("Found %s subnets for the specified zones.", len(subnets))
        return subnets
    except ClientError as err:

```

```

        log.error(
            f"Failed to retrieve subnets for VPC '{vpc_id}' in zones
{zones}."
        )
        error_code = err.response["Error"]["Code"]
        if error_code == "InvalidVpcID.NotFound":
            log.error(
                "The specified VPC ID does not exist. "
                "Please check the VPC ID and try again."
            )
        # Add more error-specific handling as needed
        log.error(f"Full error:\n\t{err}")

```

Crie uma classe que envolva ações do Elastic Load Balancing.

```

class ElasticLoadBalancerWrapper:
    """Encapsulates Elastic Load Balancing (ELB) actions."""

    def __init__(self, elb_client: boto3.client):
        """
        Initializes the LoadBalancer class with the necessary parameters.
        """
        self.elb_client = elb_client

    def create_target_group(
        self, target_group_name: str, protocol: str, port: int, vpc_id: str
    ) -> Dict[str, Any]:
        """
        Creates 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.

```
:param target_group_name: The name of the target group to create.
: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=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(f"Created load balancing target group
'{target_group_name}'.")
    return target_group
except ClientError as err:
    log.error(
        f"Couldn't create load balancing target group
'{target_group_name}'."
    )
    error_code = err.response["Error"]["Code"]

    if error_code == "DuplicateTargetGroupName":
        log.error(
            f"Target group name {target_group_name} already exists. "
            "Check if the target group already exists."
            "Consider using a different name or deleting the existing
target group if appropriate."
        )
    elif error_code == "TooManyTargetGroups":
        log.error(
            "Too many target groups exist in the account. "
            "Consider deleting unused target groups to create space for
new ones."
        )
    )
```

```

        log.error(f"Full error:\n\t{err}")

def delete_target_group(self, target_group_name) -> None:
    """
    Deletes the target group.
    """
    try:
        # Describe the target group to get its ARN
        response =
self.elb_client.describe_target_groups(Names=[target_group_name])
        tg_arn = response["TargetGroups"][0]["TargetGroupArn"]

        # Delete the target group
        self.elb_client.delete_target_group(TargetGroupArn=tg_arn)
        log.info("Deleted load balancing target group %s.",
target_group_name)

        # Use a custom waiter to wait until the target group is no longer
available
        self.wait_for_target_group_deletion(self.elb_client, tg_arn)
        log.info("Target group %s successfully deleted.", target_group_name)

    except ClientError as err:
        error_code = err.response["Error"]["Code"]
        log.error(f"Failed to delete target group '{target_group_name}'.")
        if error_code == "TargetGroupNotFound":
            log.error(
                "Load balancer target group either already deleted or never
existed. "
                "Verify the name and check that the resource exists in the
AWS Console."
            )
        elif error_code == "ResourceInUseException":
            log.error(
                "Target group still in use by another resource. "
                "Ensure that the target group is no longer associated with
any load balancers or resources.",
            )
            log.error(f"Full error:\n\t{err}")

def wait_for_target_group_deletion(
    self, elb_client, target_group_arn, max_attempts=10, delay=30
):

```

```

        for attempt in range(max_attempts):
            try:

elb_client.describe_target_groups(TargetGroupArns=[target_group_arn])
                print(
                    f"Attempt {attempt + 1}: Target group {target_group_arn}
still exists."
                )
            except ClientError as e:
                if e.response["Error"]["Code"] == "TargetGroupNotFound":
                    print(
                        f"Target group {target_group_arn} has been successfully
deleted."
                    )
                    return
                else:
                    raise
                    time.sleep(delay)
            raise TimeoutError(
                f"Target group {target_group_arn} was not deleted after {max_attempts
* delay} seconds."
            )

def create_load_balancer(
    self,
    load_balancer_name: str,
    subnet_ids: List[str],
) -> Dict[str, Any]:
    """
    Creates an Elastic Load Balancing load balancer that uses the specified
subnets
and forwards requests to the specified target group.

:param load_balancer_name: The name of the load balancer to create.
:param subnet_ids: A list of subnets to associate with the load balancer.
:return: Data about the newly created load balancer.
    """
    try:
        response = self.elb_client.create_load_balancer(
            Name=load_balancer_name, Subnets=subnet_ids
        )
        load_balancer = response["LoadBalancers"][0]
        log.info(f"Created load balancer '{load_balancer_name}'.")

```

```
        waiter = self.elb_client.get_waiter("load_balancer_available")
        log.info(
            f"Waiting for load balancer '{load_balancer_name}' to be
available..."
        )
        waiter.wait(Names=[load_balancer_name])
        log.info(f"Load balancer '{load_balancer_name}' is now available!")

    except ClientError as err:
        error_code = err.response["Error"]["Code"]
        log.error(
            f"Failed to create load balancer '{load_balancer_name}'. Error
code: {error_code}, Message: {err.response['Error']['Message']}"
        )

        if error_code == "DuplicateLoadBalancerNameException":
            log.error(
                f"A load balancer with the name '{load_balancer_name}'
already exists. "
                "Load balancer names must be unique within the AWS region. "
                "Please choose a different name and try again."
            )

        if error_code == "TooManyLoadBalancersException":
            log.error(
                "The maximum number of load balancers has been reached in
this account and region. "
                "You can delete unused load balancers or request an increase
in the service quota from AWS Support."
            )
            log.error(f"Full error:\n\t{err}")
        else:
            return load_balancer

def create_listener(
    self,
    load_balancer_name: str,
    target_group: Dict[str, Any],
) -> Dict[str, Any]:
    """
    Creates a listener for the specified load balancer that forwards requests
to the
    specified target group.
```



```

        :param load_balancer_name: The name of the load balancer to create a
listener for.
        :param target_group: An existing target group that is added as a listener
to the
                load balancer.
:return: Data about the newly created listener.
"""
try:
    # Retrieve the load balancer ARN
    load_balancer_response = self.elb_client.describe_load_balancers(
        Names=[load_balancer_name]
    )
    load_balancer_arn = load_balancer_response["LoadBalancers"][0][
        "LoadBalancerArn"
    ]

    # Create the listener
    response = self.elb_client.create_listener(
        LoadBalancerArn=load_balancer_arn,
        Protocol=target_group["Protocol"],
        Port=target_group["Port"],
        DefaultActions=[
            {
                "Type": "forward",
                "TargetGroupArn": target_group["TargetGroupArn"],
            }
        ],
    )
    log.info(
        f"Created listener to forward traffic from load balancer
'{load_balancer_name}' to target group '{target_group['TargetGroupName']}'."
    )
    return response["Listeners"][0]
except ClientError as err:
    error_code = err.response["Error"]["Code"]
    log.error(
        f"Failed to add a listener on '{load_balancer_name}' for target
group '{target_group['TargetGroupName']}'."
    )

    if error_code == "ListenerNotFoundException":
        log.error(

```

```

        f"The listener could not be found for the load balancer
        '{load_balancer_name}'. "
        "Please check the load balancer name and target group
        configuration."
    )
    if error_code == "InvalidConfigurationRequestException":
        log.error(
            f"The configuration provided for the listener on load
            balancer '{load_balancer_name}' is invalid. "
            "Please review the provided protocol, port, and target group
            settings."
        )
        log.error(f"Full error:\n\t{err}")

def delete_load_balancer(self, load_balancer_name) -> None:
    """
    Deletes a load balancer.

    :param load_balancer_name: The name of the load balancer to delete.
    """
    try:
        response = self.elb_client.describe_load_balancers(
            Names=[load_balancer_name]
        )
        lb_arn = response["LoadBalancers"][0]["LoadBalancerArn"]
        self.elb_client.delete_load_balancer(LoadBalancerArn=lb_arn)
        log.info("Deleted load balancer %s.", load_balancer_name)
        waiter = self.elb_client.get_waiter("load_balancers_deleted")
        log.info("Waiting for load balancer to be deleted...")
        waiter.wait(Names=[load_balancer_name])
    except ClientError as err:
        error_code = err.response["Error"]["Code"]
        log.error(
            f"Couldn't delete load balancer '{load_balancer_name}'. Error
            code: {error_code}, Message: {err.response['Error']['Message']}"
        )

        if error_code == "LoadBalancerNotFoundException":
            log.error(
                f"The load balancer '{load_balancer_name}' does not exist. "
                "Please check the name and try again."
            )
            log.error(f"Full error:\n\t{err}")

```

```
def get_endpoint(self, load_balancer_name) -> str:
    """
    Gets the HTTP endpoint of the load balancer.

    :return: The endpoint.
    """
    try:
        response = self.elb_client.describe_load_balancers(
            Names=[load_balancer_name]
        )
        return response["LoadBalancers"][0]["DNSName"]
    except ClientError as err:
        log.error(
            f"Couldn't get the endpoint for load balancer
{load_balancer_name}"
        )
        error_code = err.response["Error"]["Code"]
        if error_code == "LoadBalancerNotFoundException":
            log.error(
                "Verify load balancer name and ensure it exists in the AWS
console."
            )
            log.error(f"Full error:\n\t{err}")

    @staticmethod
    def verify_load_balancer_endpoint(endpoint) -> bool:
        """
        Verify this computer can successfully send a GET request to the load
balancer endpoint.

        :param endpoint: The endpoint to verify.
        :return: True if the GET request is successful, False otherwise.
        """
        retries = 3
        verified = False
        while not verified and retries > 0:
            try:
                lb_response = requests.get(f"http://{endpoint}")
                log.info(
                    "Got response %s from load balancer endpoint.",
                    lb_response.status_code,
                )
            
```

```

        if lb_response.status_code == 200:
            verified = True
        else:
            retries = 0
    except requests.exceptions.ConnectionError:
        log.info(
            "Got connection error from load balancer endpoint,
retrying..."
        )
        retries -= 1
        time.sleep(10)
    return verified

def check_target_health(self, target_group_name: str) -> List[Dict[str,
Any]]:
    """
    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=[target_group_name]
        )
        health_response = self.elb_client.describe_target_health(
            TargetGroupArn=tg_response["TargetGroups"][0]["TargetGroupArn"]
        )
    except ClientError as err:
        log.error(f"Couldn't check health of {target_group_name} target(s).")
        error_code = err.response["Error"]["Code"]
        if error_code == "LoadBalancerNotFoundException":
            log.error(
                "Load balancer associated with the target group was not
found. "
                "Ensure the load balancer exists, is in the correct AWS
region, and "
                "that you have the necessary permissions to access it.",
            )
        elif error_code == "TargetGroupNotFoundException":
            log.error(
                "Target group was not found. "
                "Verify the target group name, check that it exists in the
correct region, "

```

```

        "and ensure it has not been deleted or created in a different
account.",
    )
    log.error(f"Full error:\n\t{err}")
else:
    return health_response["TargetHealthDescriptions"]

```

Crie uma classe que use o DynamoDB para simular um serviço de recomendação.

```

class RecommendationService:
    """
    Encapsulates a DynamoDB table to use as a service that recommends books,
    movies,
    and songs.
    """

    def __init__(self, table_name: str, dynamodb_client: boto3.client):
        """
        Initializes the RecommendationService class with the necessary
        parameters.

        :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

    def create(self) -> Dict[str, Any]:
        """
        Creates a DynamoDB table to use as 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.
        :raises RecommendationServiceError: If the table creation fails.

```

```

    """
    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 done.", self.table_name)
        else:
            raise RecommendationServiceError(
                self.table_name, f"ClientError when creating table: {err}."
            )
    else:
        return response

def populate(self, data_file: str) -> None:
    """
    Populates the recommendations table from a JSON file.

    :param data_file: The path to the data file.
    :raises RecommendationServiceError: If the table population fails.
    """
    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) -> None:
    """
    Deletes the recommendations table.

    :raises RecommendationServiceError: If the table deletion fails.
    """
    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}."
            )

```

Crie uma classe que envolva ações do Systems Manager.

```

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: str = "doc-example-resilient-architecture-table"
failure_response: str = "doc-example-resilient-architecture-failure-response"
health_check: str = "doc-example-resilient-architecture-health-check"

def __init__(self, table_name: str, ssm_client: boto3.client):
    """
    Initializes the ParameterHelper class with the necessary parameters.

    :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

def reset(self) -> None:
    """
    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: str, value: str) -> None:
    """
    Sets the value of a named Systems Manager parameter.

    :param name: The name of the parameter.
    :param value: The new value of the parameter.
    :raises ParameterHelperError: If the parameter value cannot be set.
    """
    try:
        self.ssm_client.put_parameter(
            Name=name, Value=value, Overwrite=True, Type="String"
        )
        log.info("Setting parameter %s to '%s'.", name, value)
    except ClientError as err:
        error_code = err.response["Error"]["Code"]
        log.error(f"Failed to set parameter {name}.")
```



```
if error_code == "ParameterLimitExceeded":
    log.error(
        "The parameter limit has been exceeded. "
        "Consider deleting unused parameters or request a limit
increase."
    )
elif error_code == "ParameterAlreadyExists":
    log.error(
        "The parameter already exists and overwrite is set to False.
"
        "Use Overwrite=True to update the parameter."
    )
log.error(f"Full error:\n\t{err}")
```

- Para API obter detalhes, consulte os tópicos a seguir em AWS SDK for Python (Boto3) Reference. API
 - [AttachLoadBalancerTargetGroups](#)
 - [CreateAutoScalingGroup](#)
 - [CreateInstanceProfile](#)
 - [CreateLaunchTemplate](#)
 - [CreateListener](#)
 - [CreateLoadBalancer](#)
 - [CreateTargetGroup](#)
 - [DeleteAutoScalingGroup](#)
 - [DeleteInstanceProfile](#)
 - [DeleteLaunchTemplate](#)
 - [DeleteLoadBalancer](#)
 - [DeleteTargetGroup](#)
 - [DescribeAutoScalingGroups](#)
 - [DescribeAvailabilityZones](#)
 - [DescribeIamInstanceProfileAssociations](#)
 - [DescribeInstances](#)
 - [DescribeLoadBalancers](#)

- [DescribeSubnets](#)
- [DescribeTargetGroups](#)
- [DescribeTargetHealth](#)
- [DescribeVpcs](#)
- [RebootInstances](#)
- [ReplacelamInstanceProfileAssociation](#)
- [TerminateInstanceInAutoScalingGroup](#)
- [UpdateAutoScalingGroup](#)

Para obter uma lista completa de guias do AWS SDK desenvolvedor e exemplos de código, consulte [Crie EC2 recursos da Amazon usando um AWS SDK](#). Este tópico também inclui informações sobre como começar e detalhes sobre SDK versões anteriores.

Monitore EC2 API solicitações da Amazon usando a Amazon CloudWatch

Você pode monitorar EC2 API solicitações da Amazon usando a Amazon CloudWatch, que coleta dados brutos e os processa em métricas legíveis, quase em tempo real. Essas métricas fornecem uma maneira simples de rastrear o uso e os resultados das EC2 API operações da Amazon ao longo do tempo. Essas informações oferecem uma perspectiva melhor sobre o desempenho de seus aplicativos da Web e permitem identificar e diagnosticar uma variedade de problemas. Você também pode definir alarmes que observem determinados limites e enviar notificações ou realizar ações específicas quando esses limites forem atingidos.

Para obter mais informações sobre CloudWatch, consulte o [Guia CloudWatch do usuário da Amazon](#).

Important

EC2APIAs métricas da Amazon são um recurso opcional. Você deve solicitar acesso a esse recurso. Para obter mais informações, consulte [the section called “Ative as EC2 API métricas da Amazon”](#).

Conteúdo

- [Ative as EC2 API métricas da Amazon](#)
- [EC2APIMétricas e dimensões da Amazon](#)
- [Retenção de dados métricos](#)
- [Monitoramento de solicitações feitas em seu nome](#)
- [Faturamento](#)
- [Trabalhando com a Amazon CloudWatch](#)

Ative as EC2 API métricas da Amazon

Use o procedimento a seguir para solicitar acesso a esse recurso para seu Conta da AWS.

Para solicitar acesso a esse recurso

1. [AWS Support Centro](#) aberto.
2. Escolha Criar caso.
3. Escolha Conta e faturamento.
4. Em Serviço, escolha Informações gerais e Introdução.
5. Em Categoria, escolha Uso AWS e serviços.
6. Selecione Próxima etapa: informações adicionais.
7. Em Subject (Assunto), insira **Request access to Amazon EC2 API metrics**.
8. Em Descrição, insira **Please grant my account access to Amazon EC2 API metrics. Related page: <https://docs.aws.amazon.com/AWSEC2/latest/APIReference/monitor.html>**. Inclua também a região onde você precisa acessar.
9. Escolha Próxima etapa: solucione ou entre em contato conosco.
10. Na guia Fale conosco, escolha seu idioma de contato preferido e método de contato.
11. Selecione Enviar.

EC2APIMétricas e dimensões da Amazon

Metrics

As EC2 API métricas da Amazon estão contidas no `AWS/EC2/API` namespace. As tabelas a seguir listam as métricas disponíveis para EC2 API solicitações da Amazon.

Métrica	Descrição
<code>ClientErrors</code>	<p>O número de API solicitações com falha causadas por erros do cliente.</p> <p>Esses erros geralmente são causados por algo que o cliente fez, como especificar um parâmetro incorreto ou inválido na solicitação ou usar uma ação ou recurso em nome de um usuário que não tem permissão para usar a ação ou o recurso.</p> <p>Unidade: contagem</p>

Métrica	Descrição
RequestLimitExceeded	<p>O número de vezes que a taxa máxima de solicitação permitida pela Amazon EC2 APIs foi excedida para sua conta.</p> <p>As EC2 API solicitações da Amazon são limitadas para ajudar a manter o desempenho do serviço. Se suas solicitações foram limitadas, você receberá o <code>Client.RequestLimitExceeded</code> erro.</p> <p>Unidade: contagem</p>
ServerErrors	<p>O número de API solicitações com falha causadas por erros internos do servidor.</p> <p>Esses erros geralmente são causados por um erro, exceção ou falha no AWS servidor.</p> <p>Unidade: contagem</p>
SuccessfulCalls	<p>O número de API solicitações bem-sucedidas.</p> <p>Unidade: contagem</p>

Dimensões

Os dados EC2 métricos da Amazon podem ser filtrados em todas as EC2 API ações. Para obter mais informações sobre dimensões, consulte [CloudWatch Conceitos da Amazon](#).

Retenção de dados métricos

EC2APIAs métricas da Amazon são enviadas CloudWatch em intervalos de 1 minuto. CloudWatch retém os dados métricos da seguinte forma:

- Pontos de dados com um período de 60 segundos (1 minuto) ficam disponíveis por 15 dias.
- Os pontos de dados com um período de 300 segundos (5 minutos) estão disponíveis por 63 dias.

- Os pontos de dados com um período de 3600 segundos (1 hora) estão disponíveis por 455 dias (15 meses).

Monitoramento de solicitações feitas em seu nome

APIsolicitações feitas por AWS serviços em seu nome, como solicitações feitas por funções vinculadas a serviços, não contam para seus limites de API limitação e não enviam métricas para a Amazon CloudWatch para sua conta. Essas solicitações não podem ser monitoradas usando CloudWatch.

APIsolicitações feitas em seu nome por provedores de serviços terceirizados contam para seus limites de API limitação e enviam métricas para a Amazon CloudWatch para sua conta. Essas solicitações podem ser monitoradas usando CloudWatch.

Faturamento

Aplicam-se CloudWatch preços e encargos padrão. Nenhuma cobrança adicional é aplicada pelo uso das EC2 API métricas da Amazon. Para obter mais informações, consulte [Amazon CloudWatch Pricing](#).

Trabalhando com a Amazon CloudWatch

Sumário

- [Visualizando CloudWatch métricas](#)
- [Criação de CloudWatch alarmes](#)

Visualizando CloudWatch métricas

Use o procedimento a seguir para visualizar as EC2 API métricas da Amazon.

Pré-requisito

Você deve habilitar o acesso às EC2 API métricas da Amazon para sua conta. Para obter mais informações, consulte [the section called “Ative as EC2 API métricas da Amazon”](#).

Para visualizar as EC2 API métricas da Amazon usando o console

1. Abra o CloudWatch console em <https://console.aws.amazon.com/cloudwatch/>.

2. No painel de navegação, escolha Métricas, Todas as métricas.
3. Na guia Procurar, escolha o namespace EC2/API/metric.
4. Para visualizar as métricas, selecione a dimensão da métrica.

Para visualizar as EC2 API métricas da Amazon usando a linha de comando

Use um dos seguintes comandos:

- [métricas de lista](#) (AWS CLI)

```
aws cloudwatch list-metrics --namespace "AWS/EC2/API"
```

- [Obtenha- CWMetricList](#) (AWS Tools for Windows PowerShell)

```
Get-CWMetricList -Namespace "AWS/EC2/API"
```

Criação de CloudWatch alarmes

Você pode criar um CloudWatch alarme que envia uma SNS mensagem da Amazon quando o alarme muda de estado. Um alarme observa uma única métrica por um período tempo que você especifica. Ele envia uma notificação para um SNS tópico com base no valor da métrica em relação a um determinado limite em vários períodos de tempo.

Por exemplo, você pode criar um alarme que monitore o número de DescribeInstances API solicitações que falham devido a erros do lado do servidor. O alarme a seguir envia uma notificação por e-mail quando o número de falhas de DescribeInstances API solicitação atinge um limite de 10 erros do lado do servidor durante um período de 5 minutos.

Pré-requisito

Você deve habilitar o acesso às EC2 API métricas da Amazon para sua conta. Para obter mais informações, consulte [the section called “Ative as EC2 API métricas da Amazon”](#).

Para criar um alarme para erros do servidor de EC2 DescribeInstances API solicitações da Amazon

1. Abra o CloudWatch console em <https://console.aws.amazon.com/cloudwatch/>.
2. No painel de navegação, escolha Alarms (Alarmes), All alarms (Todos os alarmes).
3. Selecione Criar alarme.

4. Escolha Seleccionar métrica e especifique o seguinte:
 - a. Escolha EC2/API.
 - b. Escolha métricas por ação.
 - c. Marque a caixa de seleção ao lado da DescribeInstances que está na mesma linha do nome da ServerErrors métrica.
 - d. Escolha Seleccionar métrica.
5. A página Specify metric and conditions (Especificar métrica e condições) será exibida, mostrando um gráfico e outras informações sobre a métrica e a estatística que você selecionou.
 - a. Em Métrica, especifique o seguinte:
 - i. Em Estatística, selecione Soma.
 - ii. Em Período, verifique se 5 minutos estão selecionados.
 - b. Em Conditions (Condições), especifique o seguinte:
 - i. Em Tipo de limite, escolha Estático.
 - ii. Para Sempre que ServerErrors for, escolha Maior/Igual >=.
 - iii. Por mais de... , insira 10.
 - c. Escolha Próximo.
6. A página Configure actions (Configurar ações) é exibida.
 - Em Notificação, especifique o seguinte:
 - i. Para o gatilho do estado de alarme, escolha Em alarme.
 - ii. Em Seleccionar um SNS tópico, escolha Seleccionar um SNS tópico existente ou Criar novo tópico e preencha os campos obrigatórios para a notificação.
 - iii. Escolha Próximo.
7. A página Adicionar nome e descrição é exibida.
 - a. Em Nome do alarme, insira um nome para o alarme. O nome deve conter somente ASCII caracteres.
 - b. Em Descrição do alarme, insira uma descrição opcional para seu alarme.
 - c. Escolha Próximo.
8. A página Visualizar e criar é exibida. Verifique se as informações estão corretas e escolha Criar alarme.

Para obter mais informações, consulte [Usando CloudWatch alarmes da Amazon](#) no Guia do CloudWatch usuário da Amazon.

As traduções são geradas por tradução automática. Em caso de conflito entre o conteúdo da tradução e da versão original em inglês, a versão em inglês prevalecerá.