



Guía para desarrolladores

# Amazon Elastic Compute Cloud



# Amazon Elastic Compute Cloud: Guía para desarrolladores

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

Las marcas comerciales y la imagen comercial de Amazon no se pueden utilizar en relación con ningún producto o servicio que no sea de Amazon de ninguna manera que pueda causar confusión entre los clientes y que menosprecie o desacredite a Amazon. Todas las demás marcas registradas que no son propiedad de Amazon son propiedad de sus respectivos propietarios, que pueden o no estar afiliados, conectados o patrocinados por Amazon.

---

# Table of Contents

Acceso programático a Amazon EC2 .....	1
Puntos de conexión de servicio .....	1
Puntos de conexión IPv4 .....	7
Puntos de conexión de doble pila (IPv4 e IPv6) .....	7
Especificación de puntos de conexión .....	8
Consistencia final .....	10
Idempotencia .....	12
Idempotencia en Amazon EC2 .....	12
RunInstances idempotencia .....	16
Ejemplos .....	17
Vuelva a intentar las recomendaciones para las solicitudes idempotentes .....	19
Limitación de solicitudes de la API .....	20
Cómo se aplica la limitación .....	20
Límites de limitación .....	22
Supervise la regulación de las API .....	29
Reintentos y retrocesos exponenciales .....	29
Solicitar un aumento de límite de .....	30
Utilizando el AWS CLI .....	32
Obtenga más información sobre AWS CLI .....	32
Utilizando AWS CloudFormation .....	33
Amazon EC2 y plantillas AWS CloudFormation .....	33
Recursos para Amazon EC2 .....	33
Obtenga más información sobre AWS CloudFormation .....	37
Uso de un SDK AWS .....	38
Ejemplos de código para la API de Amazon EC2 .....	38
Obtén más información sobre los SDK AWS .....	38
API de bajo nivel para Amazon EC2 .....	39
Console-to-Code .....	40
Funcionamiento .....	40
Limitaciones .....	41
Regiones admitidas .....	41
Formatos de código admitidos .....	41
Acciones conservadas .....	41
Tabla de acciones registradas .....	42

Uso de Console-to-Code .....	42
Ejemplos de código .....	45
Acciones .....	61
AcceptVpcPeeringConnection .....	67
AllocateAddress .....	69
AllocateHosts .....	81
AssignPrivateIpAddresses .....	84
AssociateAddress .....	85
AssociateDhcpOptions .....	99
AssociateRouteTable .....	100
AttachInternetGateway .....	101
AttachNetworkInterface .....	102
AttachVolume .....	104
AttachVpnGateway .....	105
AuthorizeSecurityGroupEgress .....	106
AuthorizeSecurityGroupIngress .....	109
CancelCapacityReservation .....	129
CancelImportTask .....	131
CancelSpotFleetRequests .....	132
CancelSpotInstanceRequests .....	134
ConfirmProductInstance .....	135
CopyImage .....	137
CopySnapshot .....	139
CreateCapacityReservation .....	141
CreateCustomerGateway .....	144
CreateDhcpOptions .....	146
CreateFlowLogs .....	148
CreateImage .....	151
CreateInstanceExportTask .....	154
CreateInternetGateway .....	155
CreateKeyPair .....	157
CreateLaunchTemplate .....	172
CreateNetworkAcl .....	181
CreateNetworkAclEntry .....	183
CreateNetworkInterface .....	184
CreatePlacementGroup .....	190

CreateRoute .....	191
CreateRouteTable .....	192
CreateSecurityGroup .....	197
CreateSnapshot .....	218
CreateSpotDatafeedSubscription .....	221
CreateSubnet .....	222
CreateTags .....	230
CreateVolume .....	233
CreateVpc .....	237
CreateVpcEndpoint .....	245
CreateVpnConnection .....	249
CreateVpnConnectionRoute .....	255
CreateVpnGateway .....	256
DeleteCustomerGateway .....	257
DeleteDhcpOptions .....	259
DeleteFlowLogs .....	260
DeleteInternetGateway .....	261
DeleteKeyPair .....	262
DeleteLaunchTemplate .....	273
DeleteNetworkAcl .....	277
DeleteNetworkAclEntry .....	278
DeleteNetworkInterface .....	280
DeletePlacementGroup .....	281
DeleteRoute .....	282
DeleteRouteTable .....	283
DeleteSecurityGroup .....	284
DeleteSnapshot .....	294
DeleteSpotDatafeedSubscription .....	296
DeleteSubnet .....	297
DeleteTags .....	298
DeleteVolume .....	300
DeleteVpc .....	301
DeleteVpnConnection .....	303
DeleteVpnConnectionRoute .....	304
DeleteVpnGateway .....	305
DeregisterImage .....	306

DescribeAccountAttributes .....	307
DescribeAddresses .....	310
DescribeAvailabilityZones .....	319
DescribeBundleTasks .....	327
DescribeCapacityReservations .....	328
DescribeCustomerGateways .....	331
DescribeDhcpOptions .....	334
DescribeFlowLogs .....	337
DescribeHostReservationOfferings .....	339
DescribeHosts .....	342
DescribeIamInstanceProfileAssociations .....	344
DescribeIdFormat .....	349
DescribeIdentityIdFormat .....	351
DescribeImageAttribute .....	353
DescribeImages .....	356
DescribeImportImageTasks .....	366
DescribeImportSnapshotTasks .....	369
DescribeInstanceAttribute .....	372
DescribeInstanceState .....	376
DescribeInstanceTypes .....	379
DescribeInstances .....	393
DescribeInternetGateways .....	422
DescribeKeyPairs .....	424
DescribeNetworkAcls .....	434
DescribeNetworkInterfaceAttribute .....	438
DescribeNetworkInterfaces .....	442
DescribePlacementGroups .....	447
DescribePrefixLists .....	448
DescribeRegions .....	450
DescribeRouteTables .....	463
DescribeScheduledInstanceAvailability .....	467
DescribeScheduledInstances .....	470
DescribeSecurityGroups .....	472
DescribeSnapshotAttribute .....	488
DescribeSnapshots .....	490
DescribeSpotDatafeedSubscription .....	496

DescribeSpotFleetInstances .....	497
DescribeSpotFleetRequestHistory .....	498
DescribeSpotFleetRequests .....	501
DescribeSpotInstanceRequests .....	505
DescribeSpotPriceHistory .....	509
DescribeSubnets .....	512
DescribeTags .....	520
DescribeVolumeAttribute .....	526
DescribeVolumeStatus .....	527
DescribeVolumes .....	530
DescribeVpcAttribute .....	534
DescribeVpcClassicLink .....	536
DescribeVpcClassicLinkDnsSupport .....	538
DescribeVpcEndpointServices .....	539
DescribeVpcEndpoints .....	544
DescribeVpcs .....	547
DescribeVpnConnections .....	555
DescribeVpnGateways .....	558
DetachInternetGateway .....	560
DetachNetworkInterface .....	561
DetachVolume .....	562
DetachVpnGateway .....	563
DisableVgwRoutePropagation .....	564
DisableVpcClassicLink .....	565
DisableVpcClassicLinkDnsSupport .....	566
DisassociateAddress .....	567
DisassociateRouteTable .....	576
EnableVgwRoutePropagation .....	576
EnableVolumeIo .....	577
EnableVpcClassicLink .....	578
EnableVpcClassicLinkDnsSupport .....	580
GetConsoleOutput .....	581
GetHostReservationPurchasePreview .....	583
GetPasswordData .....	584
ImportImage .....	586
ImportKeyPair .....	589

ImportSnapshot .....	590
ModifyCapacityReservation .....	592
ModifyHosts .....	594
ModifyIdFormat .....	595
ModifyImageAttribute .....	597
ModifyInstanceAttribute .....	599
ModifyInstanceCreditSpecification .....	603
ModifyNetworkInterfaceAttribute .....	604
ModifyReservedInstances .....	606
ModifySnapshotAttribute .....	609
ModifySpotFleetRequest .....	610
ModifySubnetAttribute .....	611
ModifyVolumeAttribute .....	613
ModifyVpcAttribute .....	614
MonitorInstances .....	616
MoveAddressToVpc .....	620
PurchaseHostReservation .....	622
PurchaseScheduledInstances .....	623
RebootInstances .....	625
RegisterImage .....	636
RejectVpcPeeringConnection .....	638
ReleaseAddress .....	639
ReleaseHosts .....	650
ReplaceIamInstanceProfileAssociation .....	651
ReplaceNetworkAclAssociation .....	658
ReplaceNetworkAclEntry .....	659
ReplaceRoute .....	660
ReplaceRouteTableAssociation .....	661
ReportInstanceStatus .....	662
RequestSpotFleet .....	663
RequestSpotInstances .....	668
ResetImageAttribute .....	673
ResetInstanceAttribute .....	674
ResetNetworkInterfaceAttribute .....	676
ResetSnapshotAttribute .....	677
RevokeSecurityGroupEgress .....	678



RevokeSecurityGroupIngress .....	680
RunInstances .....	682
RunScheduledInstances .....	704
StartInstances .....	706
StopInstances .....	722
TerminateInstances .....	739
UnassignPrivateIpAddresses .....	751
UnmonitorInstances .....	752
Escenarios .....	756
Creación y administración de un servicio resiliente .....	756
Comience a utilizar instancias .....	917
Supervise las solicitudes de API mediante CloudWatch .....	1056
Habilitar las métricas de la API de Amazon EC2 .....	1056
Métricas y dimensiones de la API de Amazon EC2 .....	1057
Métricas .....	1057
Dimensiones .....	1058
Retención de datos métricos .....	1058
Supervisar las solicitudes realizadas en su nombre .....	1059
Facturación .....	1059
Trabajando con Amazon CloudWatch .....	1059
Visualización de CloudWatch las métricas .....	1059
Creando alarmas CloudWatch .....	1060
.....	mlxiii

# Acceso programático a Amazon EC2

Puede crear y administrar sus recursos de Amazon EC2 mediante la interfaz programática AWS Management Console o una interfaz programática. Para obtener información sobre el uso de la consola Amazon EC2, consulte la Guía del usuario de [Amazon EC2](#).

## Funcionamiento

- [Puntos de enlace de Amazon EC2](#)
- [Coherencia eventual](#)
- [Idempotencia](#)
- [Solicita una limitación](#)

## Interfaces de programación

- [AWS Command Line Interface \(AWS CLI\)](#)
- [AWS CloudFormation](#)
- [AWS SDK](#)
- [API de bajo nivel](#)

## Introducción

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

## Supervisión

- [AWS CloudTrail](#)
- [Supervise las solicitudes](#)

# Puntos de enlace del servicio Amazon EC2

Un punto final es una URL que sirve como punto de entrada para un AWS servicio web. Amazon EC2 admite los siguientes tipos de puntos de conexión:

- Puntos de conexión IPv4
- Puntos de conexión de doble pila compatibles con IPv4 e IPv6
- Puntos de conexión FIPS

Al hacer una solicitud, puede especificar el punto de conexión y la región que se van a utilizar. Si no especifica un punto de conexión, se utilizará de forma predeterminada el punto de conexión IPv4. Para utilizar un tipo de punto de conexión diferente, debe especificarlo en la solicitud. Para ver ejemplos prácticos, consulte [Especificación de puntos de conexión](#).

Nombre de la región	Región	Punto de conexión	Protocolo
Este de EE. UU. (Ohio)	us-east-2	ec2.us-east-2.amazonaws.com	HTTP y HTTPS
		ec2-fips.us-east-2.amazonaws.com	HTTPS
		ec2.us-east-2.api.aws	HTTPS
Este de EE. UU. (Norte de Virginia)	us-east-1	ec2.us-east-1.amazonaws.com	HTTP y HTTPS
		ec2-fips.us-east-1.amazonaws.com	HTTPS
		ec2.us-east-1.api.aws	HTTPS
Oeste de EE. UU. (Norte de California)	us-west-1	ec2.us-west-1.amazonaws.com	HTTP y HTTPS
		ec2-fips.us-west-1.amazonaws.com	HTTPS
		ec2.us-west-1.api.aws	HTTPS
Oeste de EE. UU. (Oregón)	us-west-2	ec2.us-west-2.amazonaws.com	HTTP y HTTPS
		ec2-fips.us-west-2.amazonaws.com	HTTPS
		ec2.us-west-2.api.aws	HTTPS

Nombre de la región	Región	Punto de conexión	Protocolo
			HTTPS
África (Ciudad del Cabo)	af-south-1	ec2.af-south-1.amazonaws.com ec2.af-south-1.api.aws	HTTP y HTTPS HTTPS
Asia-Pacífico (Hong Kong)	ap-east-1	ec2.ap-east-1.amazonaws.com ec2.ap-east-1.api.aws	HTTP y HTTPS HTTPS
Asia-Pacífico (Hyderabad)	ap-south-2	ec2.ap-south-2.amazonaws.com	HTTPS
Asia-Pacífico (Yakarta)	ap-southeast-3	ec2.ap-southeast-3.amazonaws.com	HTTPS
Asia-Pacífico (Melbourne)	ap-southeast-4	ec2.ap-southeast-4.amazonaws.com	HTTPS
Asia-Pacífico (Bombay)	ap-south-1	ec2.ap-south-1.amazonaws.com ec2.ap-south-1.api.aws	HTTP y HTTPS HTTPS
Asia-Pacífico (Osaka)	ap-northeast-3	ec2.ap-northeast-3.amazonaws.com	HTTP y HTTPS

Nombre de la región	Región	Punto de conexión	Protocolo
Asia-Pacífico (Seúl)	ap-northeast-2	ec2.ap-northeast-2.amazonaws.com ec2.ap-northeast-2.api.aws	HTTP y HTTPS HTTPS
Asia-Pacífico (Singapur)	ap-southeast-1	ec2.ap-southeast-1.amazonaws.com ec2.ap-southeast-1.api.aws	HTTP y HTTPS HTTPS
Asia-Pacífico (Sídney)	ap-southeast-2	ec2.ap-southeast-2.amazonaws.com ec2.ap-southeast-2.api.aws	HTTP y HTTPS HTTPS
Asia-Pacífico (Tokio)	ap-northeast-1	ec2.ap-northeast-1.amazonaws.com ec2.ap-northeast-1.api.aws	HTTP y HTTPS HTTPS
Canadá (centro)	ca-central-1	ec2.ca-central-1.amazonaws.com ec2-fips.ca-central-1.amazonaws.com ec2.ca-central-1.api.aws	HTTP y HTTPS HTTPS HTTPS
Oeste de Canadá (Calgary)	ca-west-1	ec2.ca-west-1.amazonaws.com ec2-fips.ca-west-1.amazonaws.com	HTTPS HTTPS
Europa (Fráncfort)	eu-central-1	ec2.eu-central-1.amazonaws.com ec2.eu-central-1.api.aws	HTTP y HTTPS HTTPS

Nombre de la región	Región	Punto de conexión	Protocolo
Europa (Irlanda)	eu-west-1	ec2.eu-west-1.amazonaws.com ec2.eu-west-1.api.aws	HTTP y HTTPS HTTPS
Europa (Londres)	eu-west-2	ec2.eu-west-2.amazonaws.com ec2.eu-west-2.api.aws	HTTP y HTTPS HTTPS
Europa (Milán)	eu-south-1	ec2.eu-south-1.amazonaws.com ec2.eu-south-1.api.aws	HTTP y HTTPS HTTPS
Europa (París)	eu-west-3	ec2.eu-west-3.amazonaws.com ec2.eu-west-3.api.aws	HTTP y HTTPS HTTPS
Europa (España)	eu-south-2	ec2.eu-south-2.amazonaws.com	HTTPS
Europa (Estocolmo)	eu-north-1	ec2.eu-north-1.amazonaws.com ec2.eu-north-1.api.aws	HTTP y HTTPS HTTPS
Europa (Zúrich)	eu-central-2	ec2.eu-central-2.amazonaws.com	HTTPS
Israel (Tel Aviv)	il-central-1	ec2.il-central-1.amazonaws.com	HTTPS

Nombre de la región	Región	Punto de conexión	Protocolo
Medio Oriente (Baréin)	me-south-1	ec2.me-south-1.amazonaws.com ec2.me-south-1.api.aws	HTTP y HTTPS HTTPS
Medio Oriente (EAU)	me-central-1	ec2.me-central-1.amazonaws.com	HTTPS
América del Sur (São Paulo)	sa-east-1	ec2.sa-east-1.amazonaws.com ec2.sa-east-1.api.aws	HTTP y HTTPS HTTPS
AWS GovCloud (Este de EE. UU.)	us-gov-east-1	ec2.us-gov-east-1.amazonaws.com ec2.us-gov-east-1.api.aws	HTTPS HTTPS
AWS GovCloud (Estados Unidos-Oeste)	us-gov-west-1	ec2.us-gov-west-1.amazonaws.com ec2.us-gov-west-1.api.aws	HTTPS HTTPS

Para obtener más información sobre las regiones, consulte [Regiones y zonas de disponibilidad](#) en la Guía del usuario de Amazon EC2. Para obtener una lista de los puntos de enlace de Amazon EC2, [consulte Regiones y puntos de enlace](#) en la Referencia general de Amazon Web Services.

## Temas

- [Puntos de conexión IPv4](#)
- [Puntos de conexión de doble pila \(IPv4 e IPv6\)](#)
- [Especificación de puntos de conexión](#)

Para obtener más información acerca de los puntos de conexión FIPS, consulte [FIPS endpoints](#) (Puntos de conexión FIPS) en la Referencia general de Amazon Web Services.

## Puntos de conexión IPv4

Los puntos de conexión IPv4 solo son compatibles con el tráfico IPv4. Los puntos de conexión IPv4 están disponibles para todas las regiones.

Si especifica el punto de conexión general `ec2.amazonaws.com`, utilizamos el punto de conexión para `us-east-1`. Para utilizar una región diferente, especifique su punto de conexión asociado. Por ejemplo, si especifica `ec2.us-east-2.amazonaws.com` como punto de conexión, dirigimos su solicitud al punto de conexión `us-east-2`.

Los nombres de puntos de conexión IPv4 utilizan la siguiente convención de nomenclatura:

- `service.region.amazonaws.com`

Por ejemplo, el nombre del punto de conexión IPv4 para la región `eu-west-1` es `ec2.eu-west-1.amazonaws.com`. Para obtener una lista de los puntos de enlace de Amazon EC2, [consulte Regiones y puntos de enlace](#) en la Referencia general de Amazon Web Services.

## Puntos de conexión de doble pila (IPv4 e IPv6)

Los puntos de conexión de doble pila admiten tráfico IPv4 e IPv6. Los puntos de enlace de doble pila solo están disponibles en las siguientes regiones:

- `us-east-1`—EE. UU. Este (norte de Virginia)
- `us-east-2`—EE.UU. Este (Ohio)
- `us-west-2`—Oeste de EE. UU. (Oregón)
- `eu-west-1`— Europa (Irlanda)
- `ap-south-1`— Asia-Pacífico (Bombay)
- `sa-east-1`— Sudamérica (São Paulo)
- `us-gov-east-1`—AWS GovCloud (Este de EE. UU.)
- `us-gov-west-1`—AWS GovCloud (EEUU-Oeste)

Cuando hace una solicitud a un punto de conexión de doble pila, la URL del punto de conexión se resuelve en una dirección IPv6 o IPv4, según el protocolo que utilicen la red y el cliente.



Amazon EC2 solo admite puntos de enlace regionales de doble pila, lo que significa que debe especificar la región como parte del nombre del punto de enlace. Los nombres de puntos de conexión de doble pila utilizan la siguiente convención de nomenclatura:

- `ec2.region.api.aws`

Por ejemplo, el nombre del punto de conexión de doble pila para la región `eu-west-1` es `ec2.eu-west-1.api.aws`. Para obtener una lista de los puntos de enlace de Amazon EC2, [consulte Regiones y puntos de enlace](#) en la Referencia general de Amazon Web Services.

## Especificación de puntos de conexión

En esta sección, se proporcionan algunos ejemplos de cómo especificar un punto de conexión al hacer una solicitud.

### AWS CLI

Los siguientes ejemplos muestran cómo especificar un punto final para la `us-east-2` región mediante el AWS CLI

- Doble pila

```
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

Los siguientes ejemplos muestran cómo especificar un punto final para la `us-east-2` región mediante el AWS SDK for Java 2.x.

- Doble pila

```
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

Los siguientes ejemplos muestran cómo especificar un punto final para la eu-west-1 región mediante la versión AWS SDK for Java 1.x.

- Doble pila

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

- IPv4

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

## AWS SDK for Go

Los siguientes ejemplos muestran cómo especificar un punto final para la us-east-1 región mediante el AWS SDK for Go.

- Doble pila

```
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")
})
```

## Coherencia eventual en la API de Amazon EC2

La API de Amazon EC2 sigue un modelo de coherencia eventual, debido a la naturaleza distribuida del sistema que admite la API. Esto significa que el resultado de un comando de API que ejecute y que afecte a sus recursos de Amazon EC2 podría no estar inmediatamente visible para todos los comandos posteriores que ejecute. Debe tener esto en cuenta cuando ejecute un comando de API que siga inmediatamente a un comando de API anterior.

La coherencia eventual puede afectar a la forma en que gestionas tus recursos. Por ejemplo, si ejecuta un comando para crear un recurso, eventualmente será visible para otros comandos. Esto significa que si ejecutas un comando para modificar o describir el recurso que acabas de crear, es posible que su identificador no se haya propagado por todo el sistema y aparecerá un error al responder que el recurso no existe.

Para gestionar la coherencia final, puede hacer lo siguiente:

- Confirme el estado del recurso antes de ejecutar un comando para modificarlo. Ejecute el `Describe` comando correspondiente mediante un algoritmo de retroceso exponencial para asegurarse de que dispone de tiempo suficiente para que el comando anterior se propague por el sistema. Para ello, ejecute el `Describe` comando varias veces, empezando con un par de segundos de espera y aumentando gradualmente hasta cinco minutos.
- Añada el tiempo de espera entre los comandos siguientes, incluso si un `Describe` comando devuelve una respuesta precisa. Aplique un algoritmo de retroceso exponencial comenzando con un par de segundos de tiempo de espera y aumente gradualmente hasta unos cinco minutos de tiempo de espera.

### Ejemplos de posibles errores de coherencia


A continuación se muestran ejemplos de códigos de error que se pueden encontrar como resultado de una posible coherencia.

- `InvalidInstanceID.NotFound`

Si ejecuta correctamente el `RunInstances` comando y, a continuación, ejecuta inmediatamente otro comando con el identificador de instancia que se proporcionó en la respuesta de `RunInstances`, es posible que devuelva un `InvalidInstanceID.NotFound` error. Esto no significa que la instancia no exista.

Algunos comandos específicos que pueden verse afectados son:

- `DescribeInstances`: Para confirmar el estado real de la instancia, ejecuta este comando mediante un algoritmo de retroceso exponencial.
- `TerminateInstances`: Para confirmar el estado de la instancia, primero ejecuta el `DescribeInstances` comando con un algoritmo de retroceso exponencial.

 Important

Si se `InvalidInstanceID.NotFound` produce un error después de ejecutarla `TerminateInstances`, esto no significa que la instancia se haya cerrado o vaya a finalizar. Es posible que la instancia siga ejecutándose. Por eso es importante confirmar primero el estado de la instancia mediante `DescribeInstances`.

- `InvalidGroup.NotFound`

Si ejecuta correctamente el `CreateSecurityGroup` comando y, a continuación, ejecuta inmediatamente otro comando con el ID del grupo de seguridad que se proporcionó en la respuesta de `CreateSecurityGroup`, es posible que devuelva un `InvalidGroup.NotFound` error. Para confirmar el estado del grupo de seguridad, ejecute el `DescribeSecurityGroups` comando mediante un algoritmo de retroceso exponencial.

- `InstanceLimitExceeded`

Ha solicitado más instancias de las que permite el límite de instancias actual para el tipo de instancia especificado. Podrías alcanzar este límite de forma inesperada si lanzas y cierras instancias rápidamente, ya que las instancias canceladas se tienen en cuenta para el límite de instancias durante un tiempo después de su finalización.

# Garantizar la idempotencia en las solicitudes de API de Amazon EC2

Cuando realiza una solicitud de API de mutación, la solicitud suele devolver un resultado antes de que se hayan completado los flujos de trabajo asíncronos de la operación. También es posible que se agote el tiempo de espera de las operaciones o que surjan otros problemas con el servidor antes de que finalicen, aunque la solicitud ya haya devuelto un resultado. Esto podría dificultar la determinación de si la solicitud se ha realizado correctamente o no, y podría dar lugar a múltiples reintentos para garantizar que la operación se completa correctamente. No obstante, si la solicitud original y los reintentos posteriores se realizan correctamente, la operación se completa varias veces. Esto significa que podrías crear más recursos de los que pretendías.

La idempotencia garantiza que una solicitud a la API no se complete más de una vez. Con una solicitud idempotente, si la solicitud original se completa correctamente, cualquier reintento posterior también se completa correctamente sin realizar ninguna otra acción. Sin embargo, el resultado puede contener información actualizada, como el estado de creación actual.

## Contenido

- [Idempotencia en Amazon EC2](#)
- [RunInstances idempotencia](#)
- [Ejemplos](#)
- [Vuelva a intentar las recomendaciones para las solicitudes idempotentes](#)

## Idempotencia en Amazon EC2

Las siguientes acciones de la API son idempotentes de forma predeterminada y no requieren configuración adicional. Los AWS CLI comandos correspondientes también admiten la idempotencia de forma predeterminada.

### Idempotente por defecto

- AssociateAddress
- CreateVpnConnection
- DisassociateAddress
- ReplaceNetworkAclAssociation

- `TerminateInstances`

Las siguientes acciones de la API admiten opcionalmente la idempotencia mediante un token de cliente. Los AWS CLI comandos correspondientes también admiten la idempotencia mediante un token de cliente. Un token de cliente es una cadena única, que distingue entre mayúsculas y minúsculas, de hasta 64 caracteres ASCII. Para realizar una solicitud de API idempotente mediante una de estas acciones, especifique un token de cliente en la solicitud. No debes reutilizar el mismo token de cliente para otras solicitudes de API. Si reintentas una solicitud que se completó correctamente con el mismo token de cliente y los mismos parámetros, el reintento se realizará correctamente sin realizar ninguna otra acción. Si reintentas realizar una solicitud correctamente con el mismo token de cliente, pero uno o varios de los parámetros son diferentes, distintos de la región o la zona de disponibilidad, el reintento fallará y se producirá un error.

#### `IdempotentParameterMismatch`

Idempotente si se utiliza un token de cliente

- `AllocateHosts`
- `AllocatePamPoolCidr`
- `AssociateClientVpnTargetNetwork`
- `AssociatePamResourceDiscovery`
- `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 idempotencia

- Regional: las solicitudes son idempotentes en cada región. Sin embargo, puedes usar la misma solicitud, incluido el mismo token de cliente, en una región diferente.
- Zonal: las solicitudes son idempotentes en cada zona de disponibilidad de una región. Por ejemplo, si especificas el mismo token de cliente en dos llamadas a `AllocateHosts` la misma región, las llamadas se realizarán correctamente si especifican valores diferentes para el parámetro `AvailabilityZone`



## RunInstances idempotencia

La acción de la [RunInstances](#) API utiliza la idempotencia regional y zonal.

El tipo de idempotencia que se utilice depende de cómo especifiques la zona de disponibilidad en tu solicitud de API. RunInstances La solicitud utiliza la idempotencia zonal en los siguientes casos:

- Si especifica explícitamente una zona de disponibilidad mediante el `AvailabilityZone` parámetro del tipo de datos de ubicación
- Si especifica implícitamente una zona de disponibilidad mediante el parámetro `SubnetId`

Si no especifica una zona de disponibilidad de forma explícita o implícita, la solicitud utiliza la idempotencia regional.

### Idempotencia zonal

La idempotencia zonal garantiza que una solicitud de RunInstances API sea idempotente en cada zona de disponibilidad de una región. Esto garantiza que una solicitud con el mismo token de cliente solo se pueda completar una vez dentro de cada zona de disponibilidad de una región. Sin embargo, el mismo token de cliente se puede usar para lanzar instancias en otras zonas de disponibilidad de la región.

Por ejemplo, si envías una solicitud idempotente para lanzar una instancia en la zona de `us-east-1a` disponibilidad y, a continuación, utilizas el mismo token de cliente en una solicitud en la zona de `us-east-1b` disponibilidad, lanzamos instancias en cada una de esas zonas de disponibilidad. Si uno o más de los parámetros son diferentes, los reintentos posteriores con el mismo token de cliente en esas zonas de disponibilidad se devuelven correctamente sin realizar ninguna otra acción o se produce un error. `IdempotentParameterMismatch`

### Idempotencia regional

La idempotencia regional garantiza que una solicitud de RunInstances API sea idempotente en una región. Esto garantiza que una solicitud con el mismo token de cliente solo se pueda completar una vez dentro de una región. Sin embargo, se puede usar exactamente la misma solicitud, con el mismo token de cliente, para lanzar instancias en una región diferente.

Por ejemplo, si envías una solicitud idempotente para lanzar una instancia en la `us-east-1` región y, a continuación, utilizas el mismo token de cliente en una solicitud en la `eu-west-1` región, lanzamos instancias en cada una de esas regiones. Si uno o más de los parámetros

son diferentes, los reintentos posteriores con el mismo token de cliente en esas regiones se devuelven correctamente sin realizar ninguna otra acción o fallan y se produce un error. `IdempotentParameterMismatch`

### Tip

Si una de las zonas de disponibilidad de la región solicitada no está disponible, `RunInstances` las solicitudes que utilizan la idempotencia regional podrían fallar. Para aprovechar las funciones de las zonas de disponibilidad que ofrece la AWS infraestructura, le recomendamos que utilice la idempotencia zonal al lanzar las instancias. `RunInstances` las solicitudes que utilizan la idempotencia zonal y se dirigen a una zona de disponibilidad disponible se aceptan aunque no haya otra zona de disponibilidad disponible en la región solicitada.

## Ejemplos

### AWS CLI ejemplos de comandos

Para hacer que un AWS CLI comando sea idempotente, añada la `--client-token` opción.

#### Ejemplo 1: Idempotencia

El siguiente comando [allocate-hosts](#) utiliza la idempotencia, ya que incluye un 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
```

#### Ejemplo 2: idempotencia regional de run-instances

El siguiente comando [run-instances](#) utiliza la idempotencia regional, ya que incluye un token de cliente, pero no especifica explícita o implícitamente una zona de disponibilidad.

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

#### Ejemplo 3: idempotencia zonal de run-instances

El siguiente comando [run-instances](#) usa la idempotencia zonal, ya que incluye un token de cliente y una zona de disponibilidad especificada de forma explícita.

```
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
```

## Ejemplos de solicitudes de API

Para que una solicitud de API sea idempotente, añada el ClientToken parámetro.

### Ejemplo 1: Idempotencia

La siguiente solicitud de [AllocateHosts](#) API utiliza la idempotencia, ya que incluye un 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
```

### Ejemplo 2: idempotencia regional RunInstances

La siguiente solicitud de [RunInstances](#) API utiliza la idempotencia regional, ya que incluye un token de cliente, pero no especifica de forma explícita o implícita una zona de disponibilidad.

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

### Ejemplo 3: idempotencia zonal RunInstances

La siguiente solicitud de [RunInstances](#) API utiliza la idempotencia zonal, ya que incluye un token de cliente y una zona de disponibilidad especificada de forma explícita.

```
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
```

## Vuelva a intentar las recomendaciones para las solicitudes idempotentes

En la siguiente tabla se muestran algunas respuestas comunes que puede obtener para solicitudes de API idempotentes y se ofrecen recomendaciones de reintento.

Respuesta	Recomendación	Comentarios
200 (OK)	No reintentar	La solicitud original se ha completado correctamente. Cualquier reintento posterior se devuelve correctamente.
<a href="#">Códigos de respuesta de la serie 400 (errores del cliente)</a>	No reintentar	<p>Hay un problema con la solicitud, uno de los siguientes:</p> <ul style="list-style-type: none"> <li>• Incluye un parámetro o una combinación de parámetros que no es válida.</li> <li>• Utiliza una acción o un recurso para el que no tiene permisos.</li> <li>• Utiliza un recurso que está en proceso de cambiar de estado.</li> </ul> <p>Si la solicitud implica un recurso que está en proceso de cambiar de estado, el reintento de la solicitud podría realizarse correctamente.</p>
<a href="#">Códigos de respuesta de la serie 500 (errores del servidor)</a>	Reintentar	El error se debe a un problema AWS del servidor y, por lo general, es transitorio. Repita

Respuesta	Recomendación	Comentarios
		la solicitud con una estrategia de retardo adecuada.

## Limitación de solicitudes para la API de Amazon EC2

Amazon EC2 limita las solicitudes de API de EC2 para cada AWS cuenta por región. Lo hacemos para mejorar el rendimiento del servicio y garantizar un uso justo para todos los clientes de Amazon EC2. La limitación garantiza que las llamadas a la API de Amazon EC2 no superen los límites máximos de solicitudes de API permitidos. Las llamadas a la API están sujetas a los límites de solicitudes independientemente de si se originan en:

- Una aplicación de terceros
- Una herramienta de línea de comandos
- La consola Amazon EC2

Si supera el límite de regulación de la API, aparece el `RequestLimitExceeded` código de error.

### Contenido

- [Cómo se aplica la limitación](#)
- [Límites de limitación](#)
- [Supervise la regulación de las API](#)
- [Reintentos y retrocesos exponenciales](#)
- [Solicitar un aumento de límite de](#)

## Cómo se aplica la limitación

Amazon EC2 utiliza el [algoritmo de depósito de fichas para implementar la limitación](#) de las API. Con este algoritmo, su cuenta tiene un bucket que contiene un número específico de tokens. La cantidad de tokens del depósito representa su límite de limitación en un segundo dado.

Amazon EC2 implementa dos tipos de limitación de API:

### Tipos de limitación de API

- [Limitación de velocidad de solicitudes](#)
- [Límite de velocidad de recursos](#)

## Limitación de velocidad de solicitudes

Con la limitación de la tasa de solicitudes, se limita la cantidad de solicitudes a la API que realiza. Cada solicitud que realice elimina un token del bucket. Por ejemplo, el tamaño del depósito para las acciones de la API que no mutan (`Describe*`) es de 100 fichas, por lo que puedes realizar hasta 100 `Describe*` solicitudes en un segundo. Si superas las 100 solicitudes en un segundo, estarás limitado y las solicitudes restantes dentro de ese segundo fallarán.

Los buckets se recargan automáticamente a una tasa fija. Si el depósito está por debajo de su capacidad máxima, se le vuelve a añadir un número determinado de fichas cada segundo hasta que alcance su capacidad máxima. Si la cubeta está llena cuando llegan las fichas de recarga, se desechan. La cubeta no puede contener más fichas que su número máximo de fichas. Por ejemplo, el tamaño del depósito para las acciones de la API que no mutan (`Describe*`) es de 100 fichas y la tasa de recarga es de 20 fichas por segundo. Si realizas 100 solicitudes de `Describe*` API en un segundo, el depósito se reduce inmediatamente a cero (0) tokens. A continuación, el depósito se rellena con 20 fichas por segundo, hasta que alcance su capacidad máxima de 100 fichas. Esto significa que la cubeta previamente vacía alcanza su capacidad máxima después de 5 segundos.

No es necesario esperar a que el depósito esté completamente lleno para poder realizar solicitudes a la API. Puede usar los tokens a medida que se vayan agregando al bucket. Si utilizas inmediatamente las fichas de recarga, la cubeta no alcanzará su capacidad máxima. Por ejemplo, el tamaño del depósito para las acciones no mutantes de la consola es de 100 fichas y la tasa de recarga es de 10 fichas por segundo. Si agotas el depósito realizando 100 solicitudes de API en un segundo, puedes seguir realizando 10 solicitudes de API por segundo. El depósito puede rellenarse hasta su capacidad máxima solo si realizas menos de 10 solicitudes de API por segundo.

## Límite de velocidad de recursos

Algunas acciones de la API, como `RunInstances` y `TerminateInstances`, tal como se describe en la tabla siguiente, utilizan la limitación de la tasa de recursos además de la limitación de la tasa de solicitudes. Estas acciones de la API tienen un depósito de fichas de recursos independiente que se agota en función de la cantidad de recursos a los que afecta la solicitud. Al igual que los grupos de fichas de solicitud, los grupos de fichas de recursos tienen un máximo que te permite rebasar y una tasa de recarga que te permite mantener un ritmo constante de solicitudes durante el tiempo que sea necesario. Si superas un límite de cupos específico para una API, incluso cuando un depósito aún no

se ha rellenado para admitir la siguiente llamada a la API, la acción de la API queda limitada aunque no hayas alcanzado el límite máximo total de la API.

Por ejemplo, el tamaño del depósito de fichas de recursos RunInstances es de 1000 fichas y la tasa de recarga es de dos fichas por segundo. Por lo tanto, puede lanzar 1000 instancias de forma inmediata mediante cualquier número de solicitudes de API, como una solicitud para 1000 instancias o cuatro solicitudes para 250 instancias. Cuando el depósito de fichas de recursos esté vacío, puedes lanzar hasta dos instancias por segundo, utilizando una solicitud para dos instancias o dos solicitudes para una instancia.

Para obtener más información, consulte [Tamaños de los depósitos de fichas de recursos y tasas de recarga](#).

## Límites de limitación

En las siguientes secciones, se describen los tamaños y las tasas de recarga del depósito de fichas de solicitud y del depósito de fichas de recurso.

### Límites

- [Solicita los tamaños de los cubos de fichas y las tasas de recarga](#)
- [Tamaños de los depósitos de fichas de recursos y tasas de recarga](#)

## Solicita los tamaños de los cubos de fichas y las tasas de recarga

Para limitar la tasa de solicitudes, las acciones de la API se agrupan en las siguientes categorías:

- **Acciones no mutantes:** acciones de la API que recuperan datos sobre los recursos. Por lo general, esta categoría incluye todas Describe\* las acciones, como DescribeRouteTablesDescribeImages, y. DescribeHosts Estas acciones de API suelen tener los límites de limitación de API más altos.
- [Acciones no mutantes ni filtradas ni paginadas: un subconjunto específico de acciones de la API que no mutan y que, cuando se invocan sin especificar la paginación ni un filtro, utilizan fichas de un grupo de fichas más pequeño.](#) Se recomienda utilizar la paginación y el filtrado para que los tokens se deduzcan del grupo de fichas estándar (más grande).
- **Acciones mutantes:** acciones de la API que crean, modifican o eliminan recursos. Por lo general, esta categoría incluye todas las acciones de la API que no están clasificadas como acciones no mutantes, como CreateVolume, yModifyHosts. DeleteSnapshot Estas acciones tienen un límite de limitación inferior al de las llamadas a la API que no mutan.

- Acciones que consumen muchos recursos: mutar las acciones de la API que requieren más tiempo y consumen más recursos. Estas acciones tienen un límite de limitación aún más bajo que las acciones mutantes. Se limitan por separado de otras acciones mutantes.
- Acciones de la consola que no mutan: acciones de la API que no mutan y que se invocan desde la consola Amazon EC2. Estas acciones de la API se limitan de forma independiente de otras acciones de la API que no mutan.
- Acciones sin categorizar: estas acciones de la API reciben sus propios tamaños de bloques de fichas y tasas de recarga, aunque, por definición, encajan en alguna de las otras categorías.

En la siguiente tabla, se muestran los tamaños de los depósitos de fichas de solicitudes y las tasas de recarga de todas las regiones. AWS

Categoría de acción de la API	Acciones	Capacidad máxima del bucket	Tasa de recarga de cubos
Acciones no mutantes	<ul style="list-style-type: none"> <li>• Describe*</li> <li>• Get*</li> </ul>	100	20
Acciones que no mutan ni filtradas ni paginadas	<ul style="list-style-type: none"> <li>• DescribeInstances</li> <li>• DescribeNetworkInterfaces</li> <li>• DescribeVolumes</li> <li>• DescribeInstanceStatus</li> <li>•</li> </ul>	50	10



Categoría de acción de la API	Acciones	Capacidad máxima del bucket	Tasa de recarga de cubos
	<p>DescribeSnapshots</p> <ul style="list-style-type: none"><li>DescribeSecurityGroups</li><li>DescribeSpotInstanceRequests</li></ul>		
Acciones mutantes	Acciones de la API que no están clasificadas como acciones no mutantes.	200	5

Categoría de acción de la API	Acciones	Capacidad máxima del bucket	Tasa de recarga de cubos
Acciones que consumen muchos recursos	<ul style="list-style-type: none"> <li>• AuthorizeSecurityGroupIngress</li> <li>• CancelSpotInstanceRequests</li> <li>• CreateKeyPair</li> <li>• RequestSpotInstances</li> <li>• RevokeSecurityGroupIngress</li> <li>• CreateVpcPeeringConnection</li> <li>• AcceptVpcPeeringConnection</li> <li>• RejectVpcPeeringConnection</li> <li>• DeleteVpcPeeringConnection</li> </ul>	50	5

Categoría de acción de la API	Acciones	Capacidad máxima del bucket	Tasa de recarga de cubos
Acciones de consola que no mutan	<ul style="list-style-type: none"> <li>Describe*</li> <li>Get*</li> </ul>	100	10
Acciones sin categoría	RunInstances	5	2
	StartInstances	5	2
	CreateVpcEndpoint	4	0.3
	ModifyVpcEndpoint	4	0.3
	DeleteVpcEndpoints	4	0.3
	AcceptVpcEndpointConnections	10	1
	RejectVpcEndpointConnections	10	1
	CreateVpcEndpointServiceConfiguration	10	1
ModifyVpcEndpointServiceConfiguration	10	1	

Categoría de acción de la API	Acciones	Capacidad máxima del bucket	Tasa de recarga de cubos
	DeleteVpc EndpointServiceConfigurations	10	1
	CreateDefaultVpc	1	1
	CreateDefaultSubnet	1	1
	MoveAddressToVpc	1	1
	RestoreAddressToClassic	1	1
	DescribeMovingAddresses	1	1
	AdvertiseByoipCidr	1	0.1
	ProvisionByoipCidr	1	0.1
	DescribeByoipCidrs	1	0,5
	DeprovisionByoipCidr	1	0.1
	WithdrawByoipCidr	1	0.1

Categoría de acción de la API	Acciones	Capacidad máxima del bucket	Tasa de recarga de cubos
	DescribeReservedInstancesOfferings	10	10
	PurchaseReservedInstancesOffering	5	5
	DescribeSpotFleetRequests	50	3
	DescribeSpotFleetInstances	100	5
	DescribeSpotFleetRequestHistory	100	5
	AssociateEnclaveCertificateIamRole	10	1
	DisassociateEnclaveCertificateIamRole	10	1

Categoría de acción de la API	Acciones	Capacidad máxima del bucket	Tasa de recarga de cubos
	GetAssociatedEnclaveCertificateIamRoles	10	1
	GetConsoleScreenshot	5 por cada cuenta de 2 por instancia	5 por cada cuenta de 1 por instancia

## Tamaños de los depósitos de fichas de recursos y tasas de recarga

En la siguiente tabla, se muestran los tamaños de los depósitos de tokens de recursos y las tasas de recarga para las acciones de la API que utilizan la limitación de la tasa de recursos.

Acción de la API	Capacidad máxima del bucket	Tasa de recarga de baldes
RunInstances	1 000	2
TerminateInstances	1 000	20
StartInstances	1 000	2
StopInstances	1 000	20

## Supervise la regulación de las API

Puede utilizar Amazon CloudWatch para supervisar las llamadas a la API de Amazon EC2 y recopilar y realizar un seguimiento de las métricas relacionadas con la limitación de las API. También puede crear una alarma que le avise cuando esté a punto de alcanzar los límites de limitación de las API. Para obtener más información, consulte [Supervise las solicitudes de API de Amazon EC2 mediante Amazon CloudWatch](#).

## Reintentos y retrocesos exponenciales

Es posible que tu aplicación deba volver a intentar una solicitud de API. Por ejemplo:

- Para comprobar si hay alguna actualización en el estado de un recurso
- Para enumerar una gran cantidad de recursos (por ejemplo, todos los volúmenes)
- Para volver a intentar una solicitud después de que se haya producido un error de servidor (5xx) o un error de limitación

Sin embargo, en el caso de un error del cliente (4xx), debes revisar la solicitud para corregir el problema antes de volver a intentarlo.

## Cambios en el estado de los recursos

Antes de empezar a sondear para comprobar si hay actualizaciones de estado, espere a que la solicitud se complete potencialmente. Por ejemplo, espera unos minutos antes de comprobar si la instancia está activa. Cuando empieces a sondear, utiliza un intervalo de espera adecuado entre las solicitudes sucesivas para reducir la tasa de solicitudes a la API. Para obtener resultados óptimos, utilice un intervalo de suspensión creciente o variable.

Como alternativa, puedes usar Amazon EventBridge para que te notifique el estado de algunos recursos. Por ejemplo, puede usar el evento de notificación de cambio de estado de una instancia de EC2 para notificarle un cambio de estado en una instancia. Para obtener más información, consulte [Automatizar el uso de Amazon EC2](#). EventBridge

## Reintentos

Cuando necesite sondear o volver a intentar una solicitud de API, le recomendamos que utilice un algoritmo de retroceso exponencial para calcular el intervalo de inactividad entre las llamadas a la API. El retardo exponencial se basa en la idea de utilizar tiempos de espera progresivamente más largos entre reintentos para las respuestas a errores consecutivos. Debe implementar un intervalo de retraso máximo, así como un número máximo de intentos. También puedes usar la fluctuación (retardo aleatorio) para evitar colisiones sucesivas. Para obtener más información, consulte [Tiempos de espera, reintentos y retardo con fluctuación](#).

Cada AWS SDK implementa una lógica de reintento automático. Para obtener más información, consulta el artículo [Comportamiento de los reintentos](#) en la Guía de referencia de AWS los SDK y las herramientas.

## Solicitar un aumento de límite de

Puedes solicitar un aumento de los límites de limitación de las API para tu. Cuenta de AWS

## Para solicitar acceso a esta función

1. [AWS Support Centro](#) abierto.
2. Elija Crear caso.
3. Elija Cuenta y facturación.
4. Para el servicio, selecciona Información general y Cómo empezar.
5. En Categoría, selecciona Uso AWS y servicios.
6. Elija Siguiente paso: información adicional.
7. En Subject (Asunto), escriba **Request an increase in my Amazon EC2 API throttling limits**.
8. En Descripción, escriba **Please increase the API throttling limits for my account. Related page: <https://docs.aws.amazon.com/AWSEC2/latest/APIReference/throttling.html>**. Incluya también la información siguiente:
  - Una descripción del caso de uso.
  - Las regiones en las que necesita un aumento.
  - El intervalo de una hora, en UTC, en el que se produjo el pico de aceleración o uso (para calcular el nuevo límite de regulación).
9. Elija Siguiente paso: Resuelva ahora o póngase en contacto con nosotros.
10. En la pestaña Contactar con nosotros, elige el idioma y el método de contacto que prefieras.
11. Seleccione Submit (Enviar).



# Cree recursos de Amazon EC2 mediante AWS CLI

Puede crear y administrar sus recursos de Amazon EC2 mediante AWS Command Line Interface (AWS CLI) en un shell de línea de comandos. AWS CLI Proporciona acceso directo a las API para Servicios de AWS, por ejemplo, Amazon EC2.

Para ver la sintaxis y los ejemplos de los comandos de Amazon EC2, consulte [ec2](#) en la AWS CLI Referencia de comandos. También puedes encontrar estos ejemplos en [aws-cli/awscli/examples/ec2](#) en github.

## Obtenga más información sobre AWS CLI

Para obtener más información sobre el AWS CLI, consulte los siguientes recursos:

- [AWS Command Line Interface](#)
- [AWS Command Line Interface Guía del usuario de la versión 2](#)
- [AWS Command Line Interface Guía del usuario para la versión 1](#)

# Cree recursos de Amazon EC2 mediante AWS CloudFormation

Amazon EC2 está integrado con AWS CloudFormation un servicio que le ayuda a modelar y configurar sus AWS recursos para que pueda dedicar menos tiempo a crear y administrar sus recursos e infraestructura. Usted crea una plantilla que describe los AWS recursos que necesita (como instancias y subredes) y AWS CloudFormation aprovisiona y configura esos recursos por usted.

Cuando la utilice AWS CloudFormation, podrá reutilizar la plantilla para configurar los recursos de Amazon EC2 de forma coherente y repetida. Describa sus recursos una vez y, a continuación, aprovisiona los mismos recursos una y otra vez en varias Cuentas de AWS regiones.

## Amazon EC2 y plantillas AWS CloudFormation

[Para aprovisionar y configurar recursos para Amazon EC2 y servicios relacionados, debe conocer AWS CloudFormation las plantillas.](#) Las plantillas son archivos de texto con formato JSON o YAML. Estas plantillas describen los recursos que aprovisionará en sus AWS CloudFormation pilas. Si no estás familiarizado con JSON o YAML, puedes usar AWS CloudFormation Designer para ayudarte a empezar con AWS CloudFormation las plantillas. Para obtener más información, consulta [¿Qué es AWS CloudFormation Designer?](#) en la Guía AWS CloudFormation del usuario.

## Recursos para Amazon EC2

### Recursos de computación

- [AWS::EC2::CapacityReservation](#)
- [AWS::EC2::CapacityReservationFlota](#)
- [AWS::EC2::EC2Fleet](#)
- [AWS::EC2::EC2Fleet](#)
- [AWS::EC2::Host](#)
- [AWS::EC2::Instance](#)
- [AWS::EC2::InstanceConnectPunto final](#)
- [AWS::EC2::LaunchTemplate](#)

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

## Recursos de red

- [AWS::EC2::CarrierGateway](#)
- [AWS::EC2::ClientVpnAuthorizationRule](#)
- [AWS::EC2::ClientVpnPunto final](#)
- [AWS::EC2::ClientVpnRuta](#)
- [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::LocalGatewayRuta](#)
- [AWS::EC2::LocalGatewayRouteTable](#)
- [AWS::EC2::LocalGatewayRouteTableVirtualInterfaceGroupAssociation](#)
- [AWS::EC2::LocalGatewayRouteTableAsociación VPC](#)
- [AWS::EC2::NatGateway](#)
- [AWS::EC2::NetworkInterface](#)

- [AWS::EC2::NetworkInsightsAccessScope](#)
- [AWS::EC2::NetworkInsightsAccessScopeAnalysis](#)
- [AWS::EC2::NetworkInsightsAnálisis](#)
- [AWS::EC2::NetworkInsightsRuta](#)
- [AWS::EC2::NetworkInterfaceAdjunto](#)
- [AWS::EC2::NetworkInterfacePermiso](#)
- [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::TrafficMirrorFiltro](#)
- [AWS::EC2::TrafficMirrorFilterRule](#)
- [AWS::EC2::TrafficMirrorSesión](#)
- [AWS::EC2::TrafficMirrorObjetivo](#)
- [AWS::EC2::TransitGateway](#)
- [AWS::EC2::TransitGatewayAdjunto](#)
- [AWS::EC2::TransitGatewayConnect](#)
- [AWS::EC2::TransitGatewayMulticastDomain](#)
- [AWS::EC2::TransitGatewayMulticastDomainAssociation](#)
- [AWS::EC2::TransitGatewayMulticastGroupMember](#)
- [AWS::EC2::TransitGatewayMulticastGroupSource](#)
- [AWS::EC2::TransitGatewayPeeringAttachment](#)
- [AWS::EC2::TransitGatewayRuta](#)
- [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 seguridad

- [AWS::EC2::KeyPair](#)
- [AWS::EC2::NetworkAcl](#)
- [AWS::EC2::NetworkAclEntrada](#)
- [AWS::EC2::SecurityGroup](#)
- [AWS::EC2::SecurityGroupSalida](#)
- [AWS::EC2::SecurityGroupEntrada](#)
- [AWS::EC2::VerifiedAccessPunto final](#)
- [AWS::EC2::VerifiedAccessGrupo](#)
- [AWS::EC2::VerifiedAccessInstancia](#)
- [AWS::EC2::VerifiedAccessTrustProvider](#)

## Recursos de almacenamiento

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

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

## Obtenga más información sobre AWS CloudFormation

Para obtener más información AWS CloudFormation, consulte los siguientes recursos:

- [AWS CloudFormation](#)
- [AWS CloudFormation Guía del usuario](#)

# Cree recursos de Amazon EC2 mediante un SDK AWS

AWS proporciona kits de desarrollo de software (SDK) para muchos lenguajes de programación populares. Un SDK hace que el desarrollo sea más eficiente al proporcionar lo siguiente:

- Componentes y bibliotecas prediseñados que puede incorporar a sus aplicaciones
- Herramientas para lenguajes específicos, como compiladores y depuradores
- Firma criptográfica de solicitudes de servicio
- Solicita reintentos
- Gestión de respuestas a errores

## Ejemplos de código para la API de Amazon EC2

Los ejemplos de código proporcionados por AWS muestran cómo usar una API y realizar tareas específicas. Para ver ejemplos de la API de Amazon EC2, consulte [Ejemplos de código para Amazon EC2](#). Para ver ejemplos adicionales, consulte [Buscar ejemplos de código para los AWS SDK](#) o [aws-doc-sdk-examples](#) en GitHub.

## Obtén más información sobre los SDK AWS

Para obtener más información sobre los AWS SDK, consulta los siguientes recursos:

- [AWS Guía de referencia de SDK y herramientas](#)
- [Herramientas sobre las que construir AWS](#)
- [¿Qué es un SDK?](#)

# API de bajo nivel para Amazon EC2

La API de bajo nivel de Amazon EC2 es la interfaz de nivel de protocolo de Amazon EC2. Al utilizar la API de bajo nivel, debe formatear correctamente todas las solicitudes HTTPS y añadir una firma digital válida a cada solicitud. Para obtener más información, consulte [Realizar solicitudes a la API de Amazon EC2 en la Referencia de API](#) de Amazon EC2. Como alternativa, puede utilizar un AWS SDK, que crea y firma las solicitudes en su nombre. Para obtener más información, consulte [Uso de un SDK AWS](#).

La API de Amazon EC2 consta de acciones y tipos de datos para varios servicios. Para ver las acciones de cada servicio, consulte las siguientes páginas de la referencia de la API de Amazon EC2.

- [AWS Client VPN actions](#)
- [Acciones de Amazon EBS](#)
- [Acciones de Amazon EC2](#)
- [AWS Network Manager actions](#)
- [AWS Acciones de Nitro Enclaves](#)
- [AWS Outposts actions](#)
- [AWS PrivateLink actions](#)
- [Acciones sobre la papelera de reciclaje](#)
- [AWS Site-to-Site VPN actions](#)
- [AWS Transit Gateway actions](#)
- [Acceso verificado de AWS actions](#)
- [Acciones de importación/exportación de máquinas virtuales](#)
- [Acciones de Amazon VPC](#)
- [Acciones de IPAM de Amazon VPC](#)
- [AWS Wavelength actions](#)



# Generación de código para las acciones de la consola mediante Console-to-Code

La característica Console-to-Code se encuentra en versión preliminar para Amazon EC2 y está sujeta a cambios. Solo está disponible en la región Este de EE. UU. (Norte de Virginia).

La consola proporciona una ruta guiada para crear recursos y probar prototipos. Si quiere crear los mismos recursos a escala, necesitará un código de automatización. Console-to-Code es una característica de la consola de Amazon EC2 que puede ayudarlo a empezar a usar el código de automatización. Console-to-Code registra las acciones de la consola, incluidos los valores predeterminados y los parámetros compatibles. A continuación, utiliza la IA generativa para sugerir código en el formato que prefiera infrastructure-as-code (IaC) para las acciones que desee. Puede usar el código como punto de partida y personalizarlo para que esté listo para producción en función de su caso de uso específico.

No se aplica ningún costo adicional por el uso de Console-to-Code.

## Funcionamiento

Console-to-Code puede ayudarlo a empezar a usar el código de automatización de la siguiente manera:

1. Realiza acciones en la consola, como lanzar una instancia o habilitar la supervisión detallada.
2. Console-to-Code registra todas sus acciones, incluidos toda la configuración predeterminada y los parámetros compatibles que proporciona la consola.
3. Debe elegir las acciones que quiera usar en los scripts de automatización. Pueden ser acciones mutantes o de solo lectura (no mutantes), o ambos tipos de acciones.
4. Console-to-Code genera código en el formato deseado infrastructure-as-code (IaC), por ejemplo. TypeScript
5. Debe copiar el código para usarlo en su herramienta de desarrollo de código o descargarlo para compartirlo.
6. A continuación, utilice el código como punto de partida para los scripts de automatización. Deberá validar que el código cumpla con su intención y que los parámetros configuren sus recursos

según lo esperado. Deberá personalizar el código para que esté listo para producción en función de su caso de uso. Una vez que esté satisfecho con el código, puede usarlo en sus scripts de automatización.

Consulte las instrucciones sobre cómo usar Console-to-Code en la consola de Amazon EC2 en [Uso de Console-to-Code](#).

## Limitaciones

Al usar Console-to-Code se aplican las siguientes limitaciones.

### Regiones admitidas

Actualmente solo está disponible en la región Este de EE. UU. (Norte de Virginia).

### Formatos de código admitidos

Actualmente, Console-to-Code puede generar infrastructure-as-code (IaC) en los siguientes formatos de código:

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

### Acciones conservadas

- Sesión actual: en la tabla Acciones grabadas solo se muestran las acciones realizadas durante la sesión actual. Las acciones realizadas durante las sesiones anteriores no se conservan.
- Actualización del navegador: las acciones grabadas se pierden al actualizar la pestaña del navegador.
- Aislamiento de pestañas: la tabla Acciones grabadas es específica de la pestaña del navegador en la que se realizaron las acciones. Las acciones realizadas en una pestaña no están visibles en la tabla Acciones grabadas de otra pestaña.

## Tabla de acciones registradas

En la siguiente tabla se enumeran y describen las columnas de la tabla Acciones registradas de la consola de Console-to-Code.

Título de la columna	Descripción
Página de la consola	Página de la consola en la que se realizó la acción.
Operación	Operación de la API.
Tipo	El tipo de acción. <ul style="list-style-type: none"><li>• Mutante: acciones de la API que crean, modifican o eliminan recursos.</li><li>• Solo lectura: acciones de la API que recuperan datos sobre los recursos (por lo general, todas acciones Describe* ).</li></ul>
Comando de la CLI	Detalles sobre la acción que se realizó, incluidos los parámetros y los valores.
Hora de creación	Hora a la que se realizó la acción.

## Uso de Console-to-Code

Siga estas instrucciones para generar código con Console-to-Code en la consola de Amazon EC2.

Para ver una animación de estos pasos, consulte [Visualización de una animación: generación de código con Console-to-Code en la consola de Amazon EC2](#).

Generación de código mediante Console-to-Code

1. Abra la consola de Amazon EC2 en la región Este de EE. UU. (Norte de Virginia) en <https://console.aws.amazon.com/ec2/home?region=us-east-1>.

**Note**

Console-to-Code se encuentra en versión preliminar y actualmente solo está disponible en la región Este de EE. UU. (Norte de Virginia). Solo se registrarán las acciones que se lleven a cabo en esta región.

- Utilice la consola para crear recursos y probar prototipos. Por ejemplo, utilice la consola para configurar y lanzar instancias y habilitar la supervisión detallada.

Console-to-Code registra todas las acciones que realiza.

- En el panel de navegación izquierdo, elija Console-to-Code.
- En la tabla Acciones registradas, revise las acciones que se registraron y decida qué acciones incluir para la generación de código.
  - Use el campo de búsqueda para filtrar la tabla por una página o acción específica de la consola. Al empezar a escribir, se filtra la tabla.
  - Utilice el menú desplegable Tipo para filtrar por todas las acciones, ya sean acciones mutantes o acciones de solo lectura.

**Note**

Solo se muestran las acciones realizadas durante la sesión actual. Para obtener más información, consulte [Acciones conservadas](#).

- Seleccione la casilla situada junto a cada acción para la que necesita que se genere código.

**Note**

Se pueden seleccionar hasta 5 acciones a la vez.

- Seleccione el botón Generar código {code}.

La etiqueta del botón tiene el último formato de código seleccionado de forma predeterminada. Para seleccionar un formato de código diferente, elija la flecha situada junto al botón.

- En Revisar código, seleccione Copiar para copiar el código y usarlo en la herramienta de desarrollo o Descargar para descargar el archivo y compartirlo.

- Utilice el código como punto de partida para su `infrastructure-as-code`. Deberá personalizar el código para que esté listo para producción en función de su caso de uso específico.

### Note

Si descubres que el código no está listo para la producción, envíanos tus comentarios sobre cómo se puede mejorar (consulta el siguiente paso 9). AWS Support no podemos ayudarte con el código generado ni con el desarrollo de tu código personalizado.

- (Opcional) Elija el pulgar hacia arriba o hacia abajo para informarnos si Console-to-Code lo ayudó. Si elige el pulgar hacia abajo, puede elegir Enviar comentarios para decirnos cómo podemos mejorar el código para ayudarlo mejor.

## Visualización de una animación: generación de código con Console-to-Code en la consola de Amazon EC2

The screenshot displays the Amazon EC2 console interface. On the left is a navigation sidebar with categories like EC2 Dashboard, Instances, Images, Elastic Block Store, and Network & Security. The main content area is divided into several panels:

- Resources:** A table showing the usage of various Amazon EC2 resources in the US East (N. Virginia) Region.
 

Resources	
You are using the following Amazon EC2 resources in the US East (N. Virginia) Region:	
Instances (running)	4
Dedicated Hosts	0
Instances	5
Load balancers	0
Security groups	14
Volumes	6
Auto Scaling Groups	0
Elastic IPs	0
Key pairs	5
Placement groups	1
Snapshots	5
- Launch instance:** A panel with a "Launch instance" button and a "Migrate a server" button. Below the buttons, it states: "Note: Your instances will launch in the US East (N. Virginia) Region".
- Service health:** A panel showing the "AWS Health Dashboard" and the current region as "US East (N. Virginia)". It includes a "Zones" table:
 

Zone name	Zone ID
us-east-1a	use1-az2
- Account attributes:** A panel showing account details like "Default VPC" (vpc-92304aeb) and various settings.
- Explore AWS:** A panel with promotional messages such as "Save up to 90% on EC2 with Spot Instances" and "Amazon GuardDuty Malware Protection".

# Ejemplos de código para Amazon EC2 con SDK AWS

Los siguientes ejemplos de código muestran cómo utilizar Amazon EC2 con un kit de desarrollo de AWS software (SDK).

Las acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Mientras las acciones muestran cómo llamar a las funciones de servicio individuales, es posible ver las acciones en contexto en los escenarios relacionados y en los ejemplos entre servicios.

Los escenarios son ejemplos de código que muestran cómo llevar a cabo una tarea específica llamando a varias funciones dentro del mismo servicio.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte [Cree recursos de Amazon EC2 mediante un SDK AWS](#). En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Introducción

## Introducción a Amazon EC2

En los siguientes ejemplos de código, se muestra cómo empezar a utilizar Amazon EC2.

.NET

AWS SDK for .NET

### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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>A Task object.</returns>
static async Task Main(string[] args)
{
    // Set up dependency injection for Amazon Elastic Compute Cloud (Amazon
    EC2).
    using var host =
    Microsoft.Extensions.Hosting.Host.CreateDefaultBuilder(args)
        .ConfigureServices((_, services) =>
            services.AddAWSService<IAmazonEC2>()
                .AddTransient<EC2Wrapper>()
        )
        .Build();

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

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

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

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

- Para obtener más información sobre la API, consulta [DescribeSecurityGroups](#) la Referencia AWS SDK for .NET de la API.

## C++

### SDK para C++

#### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

#### Código para el MakeLists archivo CMake C.txt.

```
# 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 del archivo de origen 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) {
    Aws::SDKOptions options;
    // Optionally change the log level for debugging.
    // options.loggingOptions.logLevel = Utils::Logging::LogLevel::Debug;
    Aws::InitAPI(options); // Should only be called once.
    int result = 0;
    {
        Aws::Client::ClientConfiguration clientConfig;
```

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

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

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

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

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

                Aws::String typeString =

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

                Aws::String monitorString =

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

```

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

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


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

```

- Para obtener más información sobre la API, consulte la Referencia de [DescribeSecurityGroups](#) la AWS SDK for C++ API.

## Java

## SDK para Java 2.x

 Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

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

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

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

- Para obtener más información sobre la API, consulta [DescribeSecurityGroups](#) la Referencia AWS SDK for Java 2.x de la API.

## JavaScript

### SDK para JavaScript (v3)

#### Note

Hay más información. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

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

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

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

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

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

- Para obtener más información sobre la API, consulta [DescribeSecurityGroups](#) la Referencia AWS SDK for JavaScript de la API.

## Kotlin

### SDK para Kotlin

#### Note

Hay más información al respecto en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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 obtener más información sobre la API, consulta [DescribeSecurityGroups](#) la referencia sobre el AWS SDK para la API de Kotlin.

## Python

### SDK para Python (Boto3)

#### Note

Hay más información al respecto en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

```
import boto3

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

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

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

- Para obtener más información sobre la API, consulta [DescribeSecurityGroups](#) la AWS Referencia de API de SDK for Python (Boto3).

## Ejemplos de código

- [Acciones para Amazon EC2 mediante SDK AWS](#)
  - [Úselo AcceptVpcPeeringConnection con un AWS SDK o CLI](#)
  - [Úselo AllocateAddress con un AWS SDK o CLI](#)
  - [Úselo AllocateHosts con un AWS SDK o CLI](#)
  - [Úselo AssignPrivateIpAddresses con un AWS SDK o CLI](#)
  - [Úselo AssociateAddress con un AWS SDK o CLI](#)
  - [Úselo AssociateDhcpOptions con un AWS SDK o CLI](#)
  - [Úselo AssociateRouteTable con un AWS SDK o CLI](#)
  - [Úselo AttachInternetGateway con un AWS SDK o CLI](#)
  - [Úselo AttachNetworkInterface con un AWS SDK o CLI](#)

- [Úselo AttachVolume con un AWS SDK o CLI](#)
- [Úselo AttachVpnGateway con un AWS SDK o CLI](#)
- [Úselo AuthorizeSecurityGroupEgress con un AWS SDK o CLI](#)
- [Úselo AuthorizeSecurityGroupIngress con un AWS SDK o CLI](#)
- [Úselo CancelCapacityReservation con un AWS SDK o CLI](#)
- [Úselo CancellImportTask con un AWS SDK o CLI](#)
- [Úselo CancelSpotFleetRequests con un AWS SDK o CLI](#)
- [Úselo CancelSpotInstanceRequests con un AWS SDK o CLI](#)
- [Úselo ConfirmProductInstance con un AWS SDK o CLI](#)
- [Úselo CopyImage con un AWS SDK o CLI](#)
- [Úselo CopySnapshot con un AWS SDK o CLI](#)
- [Úselo CreateCapacityReservation con un AWS SDK o CLI](#)
- [Úselo CreateCustomerGateway con un AWS SDK o CLI](#)
- [Úselo CreateDhcpOptions con un AWS SDK o CLI](#)
- [Úselo CreateFlowLogs con un AWS SDK o CLI](#)
- [Úselo CreateImage con un AWS SDK o CLI](#)
- [Úselo CreateInstanceExportTask con un AWS SDK o CLI](#)
- [Úselo CreateInternetGateway con un AWS SDK o CLI](#)
- [Úselo CreateKeyPair con un AWS SDK o CLI](#)
- [Úselo CreateLaunchTemplate con un AWS SDK o CLI](#)
- [Úselo CreateNetworkAcl con un AWS SDK o CLI](#)
- [Úselo CreateNetworkAclEntry con un AWS SDK o CLI](#)
- [Úselo CreateNetworkInterface con un AWS SDK o CLI](#)
- [Úselo CreatePlacementGroup con un AWS SDK o CLI](#)
- [Úselo CreateRoute con un AWS SDK o CLI](#)
- [Úselo CreateRouteTable con un AWS SDK o CLI](#)
- [Úselo CreateSecurityGroup con un AWS SDK o CLI](#)
- [Úselo CreateSnapshot con un AWS SDK o CLI](#)
- [Úselo CreateSpotDatafeedSubscription con un AWS SDK o CLI](#)
- [Úselo CreateSubnet con un AWS SDK o CLI](#)



- [Úselo CreateTags con un AWS SDK o CLI](#)
- [Úselo CreateVolume con un AWS SDK o CLI](#)
- [Úselo CreateVpc con un AWS SDK o CLI](#)
- [Úselo CreateVpcEndpoint con un AWS SDK o CLI](#)
- [Úselo CreateVpnConnection con un AWS SDK o CLI](#)
- [Úselo CreateVpnConnectionRoute con un AWS SDK o CLI](#)
- [Úselo CreateVpnGateway con un AWS SDK o CLI](#)
- [Úselo DeleteCustomerGateway con un AWS SDK o CLI](#)
- [Úselo DeleteDhcpOptions con un AWS SDK o CLI](#)
- [Úselo DeleteFlowLogs con un AWS SDK o CLI](#)
- [Úselo DeleteInternetGateway con un AWS SDK o CLI](#)
- [Úselo DeleteKeyPair con un AWS SDK o CLI](#)
- [Úselo DeleteLaunchTemplate con un AWS SDK o CLI](#)
- [Úselo DeleteNetworkAcl con un AWS SDK o CLI](#)
- [Úselo DeleteNetworkAclEntry con un AWS SDK o CLI](#)
- [Úselo DeleteNetworkInterface con un AWS SDK o CLI](#)
- [Úselo DeletePlacementGroup con un AWS SDK o CLI](#)
- [Úselo DeleteRoute con un AWS SDK o CLI](#)
- [Úselo DeleteRouteTable con un AWS SDK o CLI](#)
- [Úselo DeleteSecurityGroup con un AWS SDK o CLI](#)
- [Úselo DeleteSnapshot con un AWS SDK o CLI](#)
- [Úselo DeleteSpotDatafeedSubscription con un AWS SDK o CLI](#)
- [Úselo DeleteSubnet con un AWS SDK o CLI](#)
- [Úselo DeleteTags con un AWS SDK o CLI](#)
- [Úselo DeleteVolume con un AWS SDK o CLI](#)
- [Úselo DeleteVpc con un AWS SDK o CLI](#)
- [Úselo DeleteVpnConnection con un AWS SDK o CLI](#)
- [Úselo DeleteVpnConnectionRoute con un AWS SDK o CLI](#)
- [Úselo DeleteVpnGateway con un AWS SDK o CLI](#)
- [Úselo DeregisterImage con un AWS SDK o CLI](#)

- [Úselo DescribeAccountAttributes con un AWS SDK o CLI](#)
- [Úselo DescribeAddresses con un AWS SDK o CLI](#)
- [Úselo DescribeAvailabilityZones con un AWS SDK o CLI](#)
- [Úselo DescribeBundleTasks con un AWS SDK o CLI](#)
- [Úselo DescribeCapacityReservations con un AWS SDK o CLI](#)
- [Úselo DescribeCustomerGateways con un AWS SDK o CLI](#)
- [Úselo DescribeDhcpOptions con un AWS SDK o CLI](#)
- [Úselo DescribeFlowLogs con un AWS SDK o CLI](#)
- [Úselo DescribeHostReservationOfferings con un AWS SDK o CLI](#)
- [Úselo DescribeHosts con un AWS SDK o CLI](#)
- [Úselo DescribelamInstanceProfileAssociations con un AWS SDK o CLI](#)
- [Úselo DescribelIdFormat con un AWS SDK o CLI](#)
- [Úselo DescribelIdentityIdFormat con un AWS SDK o CLI](#)
- [Úselo DescribelImageAttribute con un AWS SDK o CLI](#)
- [Úselo DescribelImages con un AWS SDK o CLI](#)
- [Úselo DescribelImportImageTasks con un AWS SDK o CLI](#)
- [Úselo DescribelImportSnapshotTasks con un AWS SDK o CLI](#)
- [Úselo DescribelInstanceAttribute con un AWS SDK o CLI](#)
- [Úselo DescribelInstanceStatus con un AWS SDK o CLI](#)
- [Úselo DescribelInstanceTypes con un AWS SDK o CLI](#)
- [Úselo DescribelInstances con un AWS SDK o CLI](#)
- [Úselo DescribelInternetGateways con un AWS SDK o CLI](#)
- [Úselo DescribeKeyPairs con un AWS SDK o CLI](#)
- [Úselo DescribeNetworkAcls con un AWS SDK o CLI](#)
- [Úselo DescribeNetworkInterfaceAttribute con un AWS SDK o CLI](#)
- [Úselo DescribeNetworkInterfaces con un AWS SDK o CLI](#)
- [Úselo DescribePlacementGroups con un AWS SDK o CLI](#)
- [Úselo DescribePrefixLists con un AWS SDK o CLI](#)
- [Úselo DescribeRegions con un AWS SDK o CLI](#)
- [Úselo DescribeRouteTables con un AWS SDK o CLI](#)

- [Úselo DescribeScheduledInstanceAvailability con un AWS SDK o CLI](#)
- [Úselo DescribeScheduledInstances con un AWS SDK o CLI](#)
- [Úselo DescribeSecurityGroups con un AWS SDK o CLI](#)
- [Úselo DescribeSnapshotAttribute con un AWS SDK o CLI](#)
- [Úselo DescribeSnapshots con un AWS SDK o CLI](#)
- [Úselo DescribeSpotDatafeedSubscription con un AWS SDK o CLI](#)
- [Úselo DescribeSpotFleetInstances con un AWS SDK o CLI](#)
- [Úselo DescribeSpotFleetRequestHistory con un AWS SDK o CLI](#)
- [Úselo DescribeSpotFleetRequests con un AWS SDK o CLI](#)
- [Úselo DescribeSpotInstanceRequests con un AWS SDK o CLI](#)
- [Úselo DescribeSpotPriceHistory con un AWS SDK o CLI](#)
- [Úselo DescribeSubnets con un AWS SDK o CLI](#)
- [Úselo DescribeTags con un AWS SDK o CLI](#)
- [Úselo DescribeVolumeAttribute con un AWS SDK o CLI](#)
- [Úselo DescribeVolumeStatus con un AWS SDK o CLI](#)
- [Úselo DescribeVolumes con un AWS SDK o CLI](#)
- [Úselo DescribeVpcAttribute con un AWS SDK o CLI](#)
- [Úselo DescribeVpcClassicLink con un AWS SDK o CLI](#)
- [Úselo DescribeVpcClassicLinkDnsSupport con un AWS SDK o CLI](#)
- [Úselo DescribeVpcEndpointServices con un AWS SDK o CLI](#)
- [Úselo DescribeVpcEndpoints con un AWS SDK o CLI](#)
- [Úselo DescribeVpcs con un AWS SDK o CLI](#)
- [Úselo DescribeVpnConnections con un AWS SDK o CLI](#)
- [Úselo DescribeVpnGateways con un AWS SDK o CLI](#)
- [Úselo DetachInternetGateway con un AWS SDK o CLI](#)
- [Úselo DetachNetworkInterface con un AWS SDK o CLI](#)
- [Úselo DetachVolume con un AWS SDK o CLI](#)
- [Úselo DetachVpnGateway con un AWS SDK o CLI](#)
- [Úselo DisableVgwRoutePropagation con un AWS SDK o CLI](#)
- [Úselo DisableVpcClassicLink con un AWS SDK o CLI](#)

- [Úselo DisableVpcClassicLinkDnsSupport con un AWS SDK o CLI](#)
- [Úselo DisassociateAddress con un AWS SDK o CLI](#)
- [Úselo DisassociateRouteTable con un AWS SDK o CLI](#)
- [Úselo EnableVgwRoutePropagation con un AWS SDK o CLI](#)
- [Úselo EnableVolumelo con un AWS SDK o CLI](#)
- [Úselo EnableVpcClassicLink con un AWS SDK o CLI](#)
- [Úselo EnableVpcClassicLinkDnsSupport con un AWS SDK o CLI](#)
- [Úselo GetConsoleOutput con un AWS SDK o CLI](#)
- [Úselo GetHostReservationPurchasePreview con un AWS SDK o CLI](#)
- [Úselo GetPasswordData con un AWS SDK o CLI](#)
- [Úselo ImportImage con un AWS SDK o CLI](#)
- [Úselo ImportKeyPair con un AWS SDK o CLI](#)
- [Úselo ImportSnapshot con un AWS SDK o CLI](#)
- [Úselo ModifyCapacityReservation con un AWS SDK o CLI](#)
- [Úselo ModifyHosts con un AWS SDK o CLI](#)
- [Úselo ModifyIdFormat con un AWS SDK o CLI](#)
- [Úselo ModifyImageAttribute con un AWS SDK o CLI](#)
- [Úselo ModifyInstanceAttribute con un AWS SDK o CLI](#)
- [Úselo ModifyInstanceCreditSpecification con un AWS SDK o CLI](#)
- [Úselo ModifyNetworkInterfaceAttribute con un AWS SDK o CLI](#)
- [Úselo ModifyReservedInstances con un AWS SDK o CLI](#)
- [Úselo ModifySnapshotAttribute con un AWS SDK o CLI](#)
- [Úselo ModifySpotFleetRequest con un AWS SDK o CLI](#)
- [Úselo ModifySubnetAttribute con un AWS SDK o CLI](#)
- [Úselo ModifyVolumeAttribute con un AWS SDK o CLI](#)
- [Úselo ModifyVpcAttribute con un AWS SDK o CLI](#)
- [Úselo MonitorInstances con un AWS SDK o CLI](#)
- [Úselo MoveAddressToVpc con un AWS SDK o CLI](#)
- [Úselo PurchaseHostReservation con un AWS SDK o CLI](#)
- [Úselo PurchaseScheduledInstances con un AWS SDK o CLI](#)

- [Úselo RebootInstances con un AWS SDK o CLI](#)
- [Úselo RegisterImage con un AWS SDK o CLI](#)
- [Úselo RejectVpcPeeringConnection con un AWS SDK o CLI](#)
- [Úselo ReleaseAddress con un AWS SDK o CLI](#)
- [Úselo ReleaseHosts con un AWS SDK o CLI](#)
- [Úselo ReplacelamInstanceProfileAssociation con un AWS SDK o CLI](#)
- [Úselo ReplaceNetworkAclAssociation con un AWS SDK o CLI](#)
- [Úselo ReplaceNetworkAclEntry con un AWS SDK o CLI](#)
- [Úselo ReplaceRoute con un AWS SDK o CLI](#)
- [Úselo ReplaceRouteTableAssociation con un AWS SDK o CLI](#)
- [Úselo ReportInstanceStatus con un AWS SDK o CLI](#)
- [Úselo RequestSpotFleet con un AWS SDK o CLI](#)
- [Úselo RequestSpotInstances con un AWS SDK o CLI](#)
- [Úselo ResetImageAttribute con un AWS SDK o CLI](#)
- [Úselo ResetInstanceAttribute con un AWS SDK o CLI](#)
- [Úselo ResetNetworkInterfaceAttribute con un AWS SDK o CLI](#)
- [Úselo ResetSnapshotAttribute con un AWS SDK o CLI](#)
- [Úselo RevokeSecurityGroupEgress con un AWS SDK o CLI](#)
- [Úselo RevokeSecurityGroupIngress con un AWS SDK o CLI](#)
- [Úselo RunInstances con un AWS SDK o CLI](#)
- [Úselo RunScheduledInstances con un AWS SDK o CLI](#)
- [Úselo StartInstances con un AWS SDK o CLI](#)
- [Úselo StopInstances con un AWS SDK o CLI](#)
- [Úselo TerminateInstances con un AWS SDK o CLI](#)
- [Úselo UnassignPrivateIpAddresses con un AWS SDK o CLI](#)
- [Úselo UnmonitorInstances con un AWS SDK o CLI](#)
- [Escenarios para Amazon EC2 con SDK AWS](#)
  - [Cree y gestione un servicio resiliente mediante un AWS SDK](#)
  - [Comience a utilizar las instancias de Amazon EC2 mediante un SDK AWS](#)

# Acciones para Amazon EC2 mediante SDK AWS

Los siguientes ejemplos de código muestran cómo realizar acciones individuales de Amazon EC2 con AWS los SDK. Estos fragmentos llaman a la API de Amazon EC2 y son fragmentos de código de programas más grandes que se deben ejecutar en contexto. Cada ejemplo incluye un enlace a GitHub, donde puede encontrar instrucciones para configurar y ejecutar el código.

Los siguientes ejemplos incluyen solo las acciones que se utilizan con mayor frecuencia. Para ver una lista completa, consulte la [Referencia de la API de Amazon Elastic Compute Cloud \(Amazon EC2\)](#).

## Ejemplos

- [Úselo AcceptVpcPeeringConnection con un AWS SDK o CLI](#)
- [Úselo AllocateAddress con un AWS SDK o CLI](#)
- [Úselo AllocateHosts con un AWS SDK o CLI](#)
- [Úselo AssignPrivateIpAddresses con un AWS SDK o CLI](#)
- [Úselo AssociateAddress con un AWS SDK o CLI](#)
- [Úselo AssociateDhcpOptions con un AWS SDK o CLI](#)
- [Úselo AssociateRouteTable con un AWS SDK o CLI](#)
- [Úselo AttachInternetGateway con un AWS SDK o CLI](#)
- [Úselo AttachNetworkInterface con un AWS SDK o CLI](#)
- [Úselo AttachVolume con un AWS SDK o CLI](#)
- [Úselo AttachVpnGateway con un AWS SDK o CLI](#)
- [Úselo AuthorizeSecurityGroupEgress con un AWS SDK o CLI](#)
- [Úselo AuthorizeSecurityGroupIngress con un AWS SDK o CLI](#)
- [Úselo CancelCapacityReservation con un AWS SDK o CLI](#)
- [Úselo CancelImportTask con un AWS SDK o CLI](#)
- [Úselo CancelSpotFleetRequests con un AWS SDK o CLI](#)
- [Úselo CancelSpotInstanceRequests con un AWS SDK o CLI](#)
- [Úselo ConfirmProductInstance con un AWS SDK o CLI](#)
- [Úselo CopyImage con un AWS SDK o CLI](#)
- [Úselo CopySnapshot con un AWS SDK o CLI](#)

- [Úselo CreateCapacityReservation con un AWS SDK o CLI](#)
- [Úselo CreateCustomerGateway con un AWS SDK o CLI](#)
- [Úselo CreateDhcpOptions con un AWS SDK o CLI](#)
- [Úselo CreateFlowLogs con un AWS SDK o CLI](#)
- [Úselo CreateImage con un AWS SDK o CLI](#)
- [Úselo CreateInstanceExportTask con un AWS SDK o CLI](#)
- [Úselo CreateInternetGateway con un AWS SDK o CLI](#)
- [Úselo CreateKeyPair con un AWS SDK o CLI](#)
- [Úselo CreateLaunchTemplate con un AWS SDK o CLI](#)
- [Úselo CreateNetworkAcl con un AWS SDK o CLI](#)
- [Úselo CreateNetworkAclEntry con un AWS SDK o CLI](#)
- [Úselo CreateNetworkInterface con un AWS SDK o CLI](#)
- [Úselo CreatePlacementGroup con un AWS SDK o CLI](#)
- [Úselo CreateRoute con un AWS SDK o CLI](#)
- [Úselo CreateRouteTable con un AWS SDK o CLI](#)
- [Úselo CreateSecurityGroup con un AWS SDK o CLI](#)
- [Úselo CreateSnapshot con un AWS SDK o CLI](#)
- [Úselo CreateSpotDatafeedSubscription con un AWS SDK o CLI](#)
- [Úselo CreateSubnet con un AWS SDK o CLI](#)
- [Úselo CreateTags con un AWS SDK o CLI](#)
- [Úselo CreateVolume con un AWS SDK o CLI](#)
- [Úselo CreateVpc con un AWS SDK o CLI](#)
- [Úselo CreateVpcEndpoint con un AWS SDK o CLI](#)
- [Úselo CreateVpnConnection con un AWS SDK o CLI](#)
- [Úselo CreateVpnConnectionRoute con un AWS SDK o CLI](#)
- [Úselo CreateVpnGateway con un AWS SDK o CLI](#)
- [Úselo DeleteCustomerGateway con un AWS SDK o CLI](#)
- [Úselo DeleteDhcpOptions con un AWS SDK o CLI](#)
- [Úselo DeleteFlowLogs con un AWS SDK o CLI](#)

- [Úselo DeleteInternetGateway con un AWS SDK o CLI](#)
- [Úselo DeleteKeyPair con un AWS SDK o CLI](#)
- [Úselo DeleteLaunchTemplate con un AWS SDK o CLI](#)
- [Úselo DeleteNetworkAcl con un AWS SDK o CLI](#)
- [Úselo DeleteNetworkAclEntry con un AWS SDK o CLI](#)
- [Úselo DeleteNetworkInterface con un AWS SDK o CLI](#)
- [Úselo DeletePlacementGroup con un AWS SDK o CLI](#)
- [Úselo DeleteRoute con un AWS SDK o CLI](#)
- [Úselo DeleteRouteTable con un AWS SDK o CLI](#)
- [Úselo DeleteSecurityGroup con un AWS SDK o CLI](#)
- [Úselo DeleteSnapshot con un AWS SDK o CLI](#)
- [Úselo DeleteSpotDatafeedSubscription con un AWS SDK o CLI](#)
- [Úselo DeleteSubnet con un AWS SDK o CLI](#)
- [Úselo DeleteTags con un AWS SDK o CLI](#)
- [Úselo DeleteVolume con un AWS SDK o CLI](#)
- [Úselo DeleteVpc con un AWS SDK o CLI](#)
- [Úselo DeleteVpnConnection con un AWS SDK o CLI](#)
- [Úselo DeleteVpnConnectionRoute con un AWS SDK o CLI](#)
- [Úselo DeleteVpnGateway con un AWS SDK o CLI](#)
- [Úselo DeregisterImage con un AWS SDK o CLI](#)
- [Úselo DescribeAccountAttributes con un AWS SDK o CLI](#)
- [Úselo DescribeAddresses con un AWS SDK o CLI](#)
- [Úselo DescribeAvailabilityZones con un AWS SDK o CLI](#)
- [Úselo DescribeBundleTasks con un AWS SDK o CLI](#)
- [Úselo DescribeCapacityReservations con un AWS SDK o CLI](#)
- [Úselo DescribeCustomerGateways con un AWS SDK o CLI](#)
- [Úselo DescribeDhcpOptions con un AWS SDK o CLI](#)
- [Úselo DescribeFlowLogs con un AWS SDK o CLI](#)
- [Úselo DescribeHostReservationOfferings con un AWS SDK o CLI](#)



- [Úselo DescribeHosts con un AWS SDK o CLI](#)
- [Úselo DescribelamInstanceProfileAssociations con un AWS SDK o CLI](#)
- [Úselo DescribeldFormat con un AWS SDK o CLI](#)
- [Úselo DescribelidentityIdFormat con un AWS SDK o CLI](#)
- [Úselo DescribelImageAttribute con un AWS SDK o CLI](#)
- [Úselo DescribelImages con un AWS SDK o CLI](#)
- [Úselo DescribelImportImageTasks con un AWS SDK o CLI](#)
- [Úselo DescribelImportSnapshotTasks con un AWS SDK o CLI](#)
- [Úselo DescribelInstanceAttribute con un AWS SDK o CLI](#)
- [Úselo DescribelInstanceStatus con un AWS SDK o CLI](#)
- [Úselo DescribelInstanceTypes con un AWS SDK o CLI](#)
- [Úselo DescribelInstances con un AWS SDK o CLI](#)
- [Úselo DescribelInternetGateways con un AWS SDK o CLI](#)
- [Úselo DescribeKeyPairs con un AWS SDK o CLI](#)
- [Úselo DescribeNetworkAcls con un AWS SDK o CLI](#)
- [Úselo DescribeNetworkInterfaceAttribute con un AWS SDK o CLI](#)
- [Úselo DescribeNetworkInterfaces con un AWS SDK o CLI](#)
- [Úselo DescribePlacementGroups con un AWS SDK o CLI](#)
- [Úselo DescribePrefixLists con un AWS SDK o CLI](#)
- [Úselo DescribeRegions con un AWS SDK o CLI](#)
- [Úselo DescribeRouteTables con un AWS SDK o CLI](#)
- [Úselo DescribeScheduledInstanceAvailability con un AWS SDK o CLI](#)
- [Úselo DescribeScheduledInstances con un AWS SDK o CLI](#)
- [Úselo DescribeSecurityGroups con un AWS SDK o CLI](#)
- [Úselo DescribeSnapshotAttribute con un AWS SDK o CLI](#)
- [Úselo DescribeSnapshots con un AWS SDK o CLI](#)
- [Úselo DescribeSpotDatafeedSubscription con un AWS SDK o CLI](#)
- [Úselo DescribeSpotFleetInstances con un AWS SDK o CLI](#)
- [Úselo DescribeSpotFleetRequestHistory con un AWS SDK o CLI](#)

- [Úselo DescribeSpotFleetRequests con un AWS SDK o CLI](#)
- [Úselo DescribeSpotInstanceRequests con un AWS SDK o CLI](#)
- [Úselo DescribeSpotPriceHistory con un AWS SDK o CLI](#)
- [Úselo DescribeSubnets con un AWS SDK o CLI](#)
- [Úselo DescribeTags con un AWS SDK o CLI](#)
- [Úselo DescribeVolumeAttribute con un AWS SDK o CLI](#)
- [Úselo DescribeVolumeStatus con un AWS SDK o CLI](#)
- [Úselo DescribeVolumes con un AWS SDK o CLI](#)
- [Úselo DescribeVpcAttribute con un AWS SDK o CLI](#)
- [Úselo DescribeVpcClassicLink con un AWS SDK o CLI](#)
- [Úselo DescribeVpcClassicLinkDnsSupport con un AWS SDK o CLI](#)
- [Úselo DescribeVpcEndpointServices con un AWS SDK o CLI](#)
- [Úselo DescribeVpcEndpoints con un AWS SDK o CLI](#)
- [Úselo DescribeVpcs con un AWS SDK o CLI](#)
- [Úselo DescribeVpnConnections con un AWS SDK o CLI](#)
- [Úselo DescribeVpnGateways con un AWS SDK o CLI](#)
- [Úselo DetachInternetGateway con un AWS SDK o CLI](#)
- [Úselo DetachNetworkInterface con un AWS SDK o CLI](#)
- [Úselo DetachVolume con un AWS SDK o CLI](#)
- [Úselo DetachVpnGateway con un AWS SDK o CLI](#)
- [Úselo DisableVgwRoutePropagation con un AWS SDK o CLI](#)
- [Úselo DisableVpcClassicLink con un AWS SDK o CLI](#)
- [Úselo DisableVpcClassicLinkDnsSupport con un AWS SDK o CLI](#)
- [Úselo DisassociateAddress con un AWS SDK o CLI](#)
- [Úselo DisassociateRouteTable con un AWS SDK o CLI](#)
- [Úselo EnableVgwRoutePropagation con un AWS SDK o CLI](#)
- [Úselo EnableVolumelo con un AWS SDK o CLI](#)
- [Úselo EnableVpcClassicLink con un AWS SDK o CLI](#)
- [Úselo EnableVpcClassicLinkDnsSupport con un AWS SDK o CLI](#)
- [Úselo GetConsoleOutput con un AWS SDK o CLI](#)

- [Úselo `GetHostReservationPurchasePreview` con un AWS SDK o CLI](#)
- [Úselo `GetPasswordData` con un AWS SDK o CLI](#)
- [Úselo `ImportImage` con un AWS SDK o CLI](#)
- [Úselo `ImportKeyPair` con un AWS SDK o CLI](#)
- [Úselo `ImportSnapshot` con un AWS SDK o CLI](#)
- [Úselo `ModifyCapacityReservation` con un AWS SDK o CLI](#)
- [Úselo `ModifyHosts` con un AWS SDK o CLI](#)
- [Úselo `ModifyIdFormat` con un AWS SDK o CLI](#)
- [Úselo `ModifyImageAttribute` con un AWS SDK o CLI](#)
- [Úselo `ModifyInstanceAttribute` con un AWS SDK o CLI](#)
- [Úselo `ModifyInstanceCreditSpecification` con un AWS SDK o CLI](#)
- [Úselo `ModifyNetworkInterfaceAttribute` con un AWS SDK o CLI](#)
- [Úselo `ModifyReservedInstances` con un AWS SDK o CLI](#)
- [Úselo `ModifySnapshotAttribute` con un AWS SDK o CLI](#)
- [Úselo `ModifySpotFleetRequest` con un AWS SDK o CLI](#)
- [Úselo `ModifySubnetAttribute` con un AWS SDK o CLI](#)
- [Úselo `ModifyVolumeAttribute` con un AWS SDK o CLI](#)
- [Úselo `ModifyVpcAttribute` con un AWS SDK o CLI](#)
- [Úselo `MonitorInstances` con un AWS SDK o CLI](#)
- [Úselo `MoveAddressToVpc` con un AWS SDK o CLI](#)
- [Úselo `PurchaseHostReservation` con un AWS SDK o CLI](#)
- [Úselo `PurchaseScheduledInstances` con un AWS SDK o CLI](#)
- [Úselo `RebootInstances` con un AWS SDK o CLI](#)
- [Úselo `RegisterImage` con un AWS SDK o CLI](#)
- [Úselo `RejectVpcPeeringConnection` con un AWS SDK o CLI](#)
- [Úselo `ReleaseAddress` con un AWS SDK o CLI](#)
- [Úselo `ReleaseHosts` con un AWS SDK o CLI](#)
- [Úselo `ReplacelamInstanceProfileAssociation` con un AWS SDK o CLI](#)
- [Úselo `ReplaceNetworkAclAssociation` con un AWS SDK o CLI](#)
- [Úselo `ReplaceNetworkAclEntry` con un AWS SDK o CLI](#)

- [Úselo `ReplaceRoute` con un AWS SDK o CLI](#)
- [Úselo `ReplaceRouteTableAssociation` con un AWS SDK o CLI](#)
- [Úselo `ReportInstanceState` con un AWS SDK o CLI](#)
- [Úselo `RequestSpotFleet` con un AWS SDK o CLI](#)
- [Úselo `RequestSpotInstances` con un AWS SDK o CLI](#)
- [Úselo `ResetImageAttribute` con un AWS SDK o CLI](#)
- [Úselo `ResetInstanceAttribute` con un AWS SDK o CLI](#)
- [Úselo `ResetNetworkInterfaceAttribute` con un AWS SDK o CLI](#)
- [Úselo `ResetSnapshotAttribute` con un AWS SDK o CLI](#)
- [Úselo `RevokeSecurityGroupEgress` con un AWS SDK o CLI](#)
- [Úselo `RevokeSecurityGroupIngress` con un AWS SDK o CLI](#)
- [Úselo `RunInstances` con un AWS SDK o CLI](#)
- [Úselo `RunScheduledInstances` con un AWS SDK o CLI](#)
- [Úselo `StartInstances` con un AWS SDK o CLI](#)
- [Úselo `StopInstances` con un AWS SDK o CLI](#)
- [Úselo `TerminateInstances` con un AWS SDK o CLI](#)
- [Úselo `UnassignPrivateIPAddresses` con un AWS SDK o CLI](#)
- [Úselo `UnmonitorInstances` con un AWS SDK o CLI](#)

## Úselo **`AcceptVpcPeeringConnection`** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `AcceptVpcPeeringConnection`.

CLI

AWS CLI

Para aceptar una conexión de emparejamiento de VPC

En este ejemplo, se acepta la solicitud de conexión de emparejamiento de VPC especificada.

Comando:

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

**Salida:**

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

- Para obtener más información sobre la API, consulte la Referencia [AcceptVpcPeeringConnection](#) de AWS CLI comandos.

**PowerShell****Herramientas para PowerShell**

Ejemplo 1: En este ejemplo se aprueba el pcx-1dfad234b56ff78be solicitado VpcPeeringConnectionId

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

**Salida:**

```
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 obtener información sobre [AcceptVpcPeeringConnection AWS Tools for PowerShell](#) API, consulte la referencia del cmdlet.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **AllocateAddress** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `AllocateAddress`.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en contexto en el siguiente ejemplo de código:

- [Comience a utilizar instancias](#)

.NET

AWS SDK for .NET

### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

```
/// <summary>  
/// Allocate an Elastic IP address.  
/// </summary>  
/// <returns>The allocation Id of the allocated address.</returns>  
public async Task<string> AllocateAddress()  
{  
    var request = new AllocateAddressRequest();  
  
    var response = await _amazonEC2.AllocateAddressAsync(request);  
    return response.AllocationId;  
}
```

- Para obtener más información sobre la API, consulta [AllocateAddress](#) la Referencia AWS SDK for .NET de la API.

## Bash

### AWS CLI con el script Bash

#### Note

Hay más información. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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
}

```

Las funciones de utilidad utilizadas en este ejemplo.

```

#####
# 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 obtener más información sobre la API, consulta [AllocateAddress](#) la Referencia de AWS CLI comandos.

## C++

### SDK para C++

#### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

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

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

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

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

- Para obtener más información sobre la API, consulta [AllocateAddress](#) la Referencia AWS SDK for C++ de la API.

## CLI

## AWS CLI

Ejemplo 1: Asignar una dirección IP elástica del grupo de direcciones de Amazon

En el siguiente ejemplo de `allocate-address`, se asigna una dirección IP elástica. Amazon EC2 selecciona la dirección del conjunto de direcciones de Amazon.

```
aws ec2 allocate-address
```

Salida:

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

Para obtener más información, consulte [Direcciones IP elásticas](#) en la Guía del usuario de Amazon EC2.

Ejemplo 2: Asignar una dirección IP elástica y asociarla a un grupo fronterizo de red

En el siguiente ejemplo de `allocate-address`, se asigna una dirección IP elástica y se la asocia al grupo fronterizo de red especificado.

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

Salida:

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

Para obtener más información, consulte [Direcciones IP elásticas](#) en la Guía del usuario de Amazon EC2.

Ejemplo 3: Asignar una dirección IP elástica desde un grupo de direcciones propio

En el siguiente ejemplo de `allocate-address`, se asigna una dirección IP elástica desde un grupo de direcciones que usted trajo a su cuenta de Amazon Web Services. Amazon EC2 selecciona la dirección del conjunto de direcciones.

```
aws ec2 allocate-address \  
  --public-ipv4-pool ipv4pool-ec2-1234567890abcdef0
```

Salida:

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

Para obtener más información, consulte [Direcciones IP elásticas](#) en la Guía del usuario de Amazon EC2.

- Para obtener más información sobre la API, consulta [AllocateAddress](#) la Referencia de AWS CLI comandos.

## Java

### SDK para Java 2.x

#### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

```
public static String allocateAddress(Ec2Client ec2) {  
    try {
```

```
        AllocateAddressRequest allocateRequest =
AllocateAddressRequest.builder()
        .domain(DomainType.VPC)
        .build();

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

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

- Para obtener más información sobre la API, consulta [AllocateAddress](#) la Referencia AWS SDK for Java 2.x de la API.

## JavaScript

### SDK para JavaScript (v3)

#### Note

Hay más información. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

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

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

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

    try {
        const { AllocationId, PublicIp } = await client.send(command);
        console.log("A new IP address has been allocated to your account:");
        console.log(`ID: ${AllocationId} Public IP: ${PublicIp}`);
    }
}
```

```
console.log(
  "You can view your IP addresses in the AWS Management Console for Amazon
  EC2. Look under Network & Security > Elastic IPs",
);
} catch (err) {
  console.error(err);
}
};
```

- Para obtener más información sobre la API, consulta [AllocateAddress](#) la Referencia AWS SDK for JavaScript de la API.

## Kotlin

### SDK para Kotlin

#### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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 obtener más información sobre la API, consulta [AllocateAddress](#) la referencia sobre el AWS SDK para la API de Kotlin.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: en este ejemplo se asigna una dirección IP elástica para usarla con una instancia en una VPC.

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

Salida:

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

Ejemplo 2: Este ejemplo asigna una dirección IP elástica para usarla con una instancia en EC2-Classic.

```
New-EC2Address
```

Salida:

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

- Para obtener más información sobre la API, consulte [AllocateAddress](#) la referencia de cmdlets.AWS Tools for PowerShell

## Python

### SDK para Python (Boto3)

#### Note

Hay más información al respecto. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

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

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

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

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

        :return: The newly created Elastic IP object. By default, the address is
        not
```



```

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

```

- Para obtener más información sobre la API, consulta [AllocateAddress](#) la AWS Referencia de API de SDK for Python (Boto3).

## Ruby

### SDK para Ruby

#### Note

Hay más información al respecto. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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")
  return response.allocation_id

```

```
rescue StandardError => e
  puts "Error allocating Elastic IP address: #{e.message}"
  return "Error"
end
```

- Para obtener más información sobre la API, consulta [AllocateAddress](#) la Referencia AWS SDK for Ruby de la API.

## SAP ABAP

### SDK de SAP ABAP

#### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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 obtener más información sobre la API, consulte [AllocateAddress](#) la referencia sobre la API ABAP del AWS SDK para SAP.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte [Cree recursos de Amazon EC2 mediante un SDK AWS](#). En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **AllocateHosts** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar AllocateHosts.

## CLI

## AWS CLI

## Ejemplo 1: Para asignar un host dedicado

En el siguiente `allocate-hosts` ejemplo, se asigna un único host dedicado en la zona de `eu-west-1a` disponibilidad, en el que se pueden lanzar `m5.large` instancias. De forma predeterminada, el host dedicado solo acepta el lanzamiento de instancias de destino y no admite la recuperación del host.

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

Salida:

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

## Ejemplo 2: Para asignar un host dedicado con la ubicación automática y la recuperación del host habilitadas

El siguiente `allocate-hosts` ejemplo asigna un único host dedicado en la zona de `eu-west-1a` disponibilidad con la ubicación automática y la recuperación del host habilitadas.

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

Salida:

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

```

    "h-07879acf49EXAMPLE"
  ]
}

```

### Ejemplo 3: Para asignar un host dedicado con etiquetas

El siguiente `allocate-hosts` ejemplo asigna un único host dedicado y aplica una etiqueta con una clave denominada `purpose` y un 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}'

```

Salida:

```

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

```

Para obtener más información, consulte [Asignación de hosts dedicados](#) en la Guía del usuario de Amazon Elastic Compute Cloud para instancias de Linux.

- Para obtener más información sobre la API, consulte [AllocateHosts](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: En este ejemplo se asigna un host dedicado a su cuenta para el tipo de instancia y la zona de disponibilidad determinados

```

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

```

Salida:

```
h-01e23f4cd567890f3
```

- Para obtener información sobre la API, consulte la referencia de [AllocateHosts AWS Tools for PowerShellcmdlets](#).

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **AssignPrivateIpAddresses** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `AssignPrivateIpAddresses`.

### CLI

#### AWS CLI

Para asignar una dirección IP privada secundaria específica a una interfaz de red

En este ejemplo, se asigna la dirección IP privada secundaria especificada a la interfaz de red especificada. Si el comando se ejecuta correctamente, no se muestra ningún resultado.

Comando:

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

Para asignar direcciones IP privadas secundarias que Amazon EC2 seleccione a una interfaz de red

En este ejemplo, se asignan dos direcciones IP privadas secundarias a la interfaz de red especificada. Amazon EC2 asigna automáticamente estas direcciones IP a partir de las direcciones IP disponibles en el rango de bloques CIDR de la subred a la que está asociada la interfaz de red. Si el comando se ejecuta correctamente, no se muestra ningún resultado.

Comando:

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

- Para obtener más información sobre la API, consulte [AssignPrivateIpAddresses](#) la Referencia de comandos.AWS CLI

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: en este ejemplo se asigna la dirección IP privada secundaria especificada a la interfaz de red especificada.

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

Ejemplo 2: este ejemplo crea dos direcciones IP privadas secundarias y las asigna a la interfaz de red especificada.

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

- Para obtener más información sobre la API, consulte la referencia [AssignPrivateIpAddresses](#) de AWS Tools for PowerShell cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **AssociateAddress** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar AssociateAddress.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en contexto en el siguiente ejemplo de código:

- [Comience a utilizar instancias](#)

## .NET

### AWS SDK for .NET

#### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

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

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

- Para obtener más información sobre la API, consulta [AssociateAddress](#) la Referencia AWS SDK for .NET de la API.

## Bash

### AWS CLI con el script Bash

#### Note

Hay más información. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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."
  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
}
```

Las funciones de utilidad utilizadas en este ejemplo.

```
#####
# 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 obtener más información sobre la API, consulta [AssociateAddress](#) la Referencia de AWS CLI comandos.

## C++

### SDK para C++

#### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

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

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

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

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

- Para obtener más información sobre la API, consulta [AssociateAddress](#) la Referencia AWS SDK for C++ de la API.

## CLI

### AWS CLI

#### Asociar una dirección IP elástica en EC2-Classic

En este ejemplo, se asocia una dirección IP elástica a una instancia en EC2-Classic. Si el comando se ejecuta correctamente, no se muestra ningún resultado.

Comando:

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

#### Asociar una dirección IP elástica en EC2-VPC

En este ejemplo, se asocia una dirección IP elástica a una instancia en una VPC.

Comando:

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

Salida:

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

En este ejemplo, se asocia una dirección IP elástica a una interfaz de red.

Comando:

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

En este ejemplo, se asocia una dirección IP elástica a una dirección IP privada asociada a una interfaz de red.

Comando:

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

- Para obtener más información sobre la API, consulta [AssociateAddress](#) la Referencia de AWS CLI comandos.

## Java

### SDK para Java 2.x

#### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

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

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

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

- Para obtener más información sobre la API, consulta [AssociateAddress](#) la Referencia AWS SDK for Java 2.x de la API.

## JavaScript

### SDK para JavaScript (v3)

#### Note

Hay más información. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

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

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

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

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

- Para obtener más información sobre la API, consulta [AssociateAddress](#) la Referencia AWS SDK for JavaScript de la API.

## Kotlin

### SDK para Kotlin

#### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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 obtener más información sobre la API, consulta [AssociateAddress](#) la referencia sobre el AWS SDK para la API de Kotlin.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: este ejemplo asocia la dirección IP elástica especificada a la instancia especificada en una VPC.

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

Salida:

```
eipassoc-12345678
```

Ejemplo 2: Este ejemplo asocia la dirección IP elástica especificada a la instancia especificada en EC2-Classic.

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

- Para obtener más información sobre la API, consulte la referencia del [AssociateAddress AWS Tools for PowerShell](#)cmdlet.

## Python

### SDK para Python (Boto3)

#### Note

Hay más información al respecto. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

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

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



```

        self.elastic_ip = elastic_ip

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

    def associate(self, instance):
        """
        Associates an Elastic IP address with an instance. When this association
        is created, the Elastic IP's public IP address is immediately used as the
        public IP address of the associated instance.

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

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

```

- Para obtener más información sobre la API, consulta [AssociateAddress](#) la AWS Referencia de API de SDK for Python (Boto3).

## Ruby

### SDK para Ruby

#### Note

Hay más información al respecto. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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,
  )
  return response.association_id
rescue StandardError => e
  puts "Error associating Elastic IP address with instance: #{e.message}"
  return "Error"
```

```
end
```

- Para obtener más información sobre la API, consulta [AssociateAddress](#) la Referencia AWS SDK for Ruby de la API.

## SAP ABAP

### SDK de SAP ABAP

#### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

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

- Para obtener más información sobre la API, consulte [AssociateAddress](#) la referencia sobre la API ABAP del AWS SDK para SAP.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte [Cree recursos de Amazon EC2 mediante un SDK AWS](#). En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo `AssociateDhcpOptions` con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `AssociateDhcpOptions`.

### CLI

#### AWS CLI

Para asociar un conjunto de opciones de DHCP a su VPC

Este ejemplo asocia el conjunto de opciones de DHCP especificado a la VPC especificada. Si el comando se ejecuta correctamente, no se muestra ningún resultado.

Comando:

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

Para asociar las opciones de DHCP predeterminadas configuradas a la VPC

En este ejemplo, se asocian las opciones de DHCP predeterminadas establecidas a la VPC especificada. Si el comando se ejecuta correctamente, no se muestra ningún resultado.

Comando:

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

- Para obtener más información sobre la API, consulte la Referencia [AssociateDhcpOptions](#) de AWS CLI comandos.

### PowerShell

#### Herramientas para PowerShell

Ejemplo 1: este ejemplo asocia el conjunto de opciones de DHCP especificado a la VPC especificada.

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

Ejemplo 2: en este ejemplo se asocian las opciones de DHCP predeterminadas establecidas a la VPC especificada.

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

- Para obtener más información sobre la API, consulte la referencia de [AssociateDhcpOptions AWS Tools for PowerShell](#) cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **AssociateRouteTable** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar AssociateRouteTable.

### CLI

#### AWS CLI

Para asociar una tabla de rutas a una subred

En este ejemplo, se asocia la tabla de rutas especificada a la subred especificada.

Comando:

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

Salida:

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

- Para obtener más información sobre la API, consulte [AssociateRouteTable](#) la Referencia de AWS CLI comandos.

### PowerShell

#### Herramientas para PowerShell

Ejemplo 1: Este ejemplo asocia la tabla de rutas especificada a la subred especificada.

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

Salida:

```
rtbassoc-12345678
```

- Para obtener más información sobre la API, consulte la referencia [AssociateRouteTable](#) de AWS Tools for PowerShell cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **AttachInternetGateway** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `AttachInternetGateway`.

CLI

AWS CLI

Para conectar una puerta de enlace de Internet a su VPC

En el siguiente `attach-internet-gateway` ejemplo, se adjunta la puerta de enlace de Internet especificada a la VPC específica.

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

Este comando no genera ninguna salida.

Para obtener más información, consulte [Puertas de enlace de Internet](#) en la Guía del usuario de Amazon VPC.

- Para obtener información sobre la API, consulte la Referencia [AttachInternetGateway](#) de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: en este ejemplo se adjunta la puerta de enlace de Internet especificada a la VPC especificada.

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

Ejemplo 2: en este ejemplo se crean una VPC y una puerta de enlace a Internet y, a continuación, se adjunta la puerta de enlace a Internet a la VPC.

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

- Para obtener información sobre la API, consulte la referencia de [AttachInternetGateway](#) cmdlets AWS Tools for PowerShell .

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **AttachNetworkInterface** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `AttachNetworkInterface`.

### CLI

#### AWS CLI

Ejemplo 1: Para adjuntar una interfaz de red a una instancia

En el siguiente `attach-network-interface` ejemplo, se adjunta la interfaz de red especificada a la instancia especificada.

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

**Salida:**

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

Para obtener más información, consulte [Interfases de red elásticas](#) en la Guía del usuario de Amazon EC2.

Ejemplo 2: Para conectar una interfaz de red a una instancia con varias tarjetas de red

En el siguiente `attach-network-interface` ejemplo, se conecta la interfaz de red especificada a la instancia y a la tarjeta de red especificadas.

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

**Salida:**

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

Para obtener más información, consulte [Interfases de red elásticas](#) en la Guía del usuario de Amazon EC2.

- Para obtener más información sobre la API, consulte [AttachNetworkInterface](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: en este ejemplo se adjunta la interfaz de red especificada a la instancia especificada.

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



**Salida:**

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

- Para obtener más información sobre la API, consulte la referencia [AttachNetworkInterface](#) de AWS Tools for PowerShell cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **AttachVolume** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `AttachVolume`.

### CLI

#### AWS CLI

Para adjuntar un volumen a una instancia

Este comando de ejemplo adjunta un volumen (`vol-1234567890abcdef0`) a una instancia (`i-01474ef662b89480`) como `/dev/sdf`.

Comando:

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

Salida:

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

- Para obtener más información sobre la API, consulta [AttachVolume](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: en este ejemplo se adjunta el volumen especificado a la instancia especificada y se expone con el nombre de dispositivo especificado.

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

#### Salida:

```
AttachTime      : 12/22/2015 1:53:58 AM
DeleteOnTermination : False
Device          : /dev/sdh
InstanceId      : i-1a2b3c4d
State           : attaching
VolumeId       : vol-12345678
```

- Para obtener más información sobre la API, consulte la referencia del [AttachVolume AWS Tools for PowerShell](#) cmdlet.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **AttachVpnGateway** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `AttachVpnGateway`.

### CLI

#### AWS CLI

Para adjuntar una puerta de enlace privada virtual a su VPC

En el siguiente `attach-vpn-gateway` ejemplo, se adjunta la puerta de enlace privada virtual especificada a la VPC especificada.

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

**Salida:**

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

- Para obtener más información sobre la API, consulte la Referencia [AttachVpnGateway](#) de AWS CLI comandos.

**PowerShell****Herramientas para PowerShell**

Ejemplo 1: en este ejemplo se adjunta la puerta de enlace privada virtual especificada a la VPC especificada.

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

**Salida:**

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

- Para obtener más información sobre la API, consulte la referencia de [AttachVpnGateway](#) [AWS Tools for PowerShell](#) cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **AuthorizeSecurityGroupEgress** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `AuthorizeSecurityGroupEgress`.

## CLI

### AWS CLI

Para agregar una regla que permita el tráfico saliente a un rango de direcciones específico

Este comando de ejemplo agrega una regla que otorga acceso a los rangos de direcciones especificados en el puerto TCP 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 agregar una regla que permita el tráfico saliente a un grupo de seguridad específico

Este comando de ejemplo agrega una regla que otorga acceso al grupo de seguridad especificado en el puerto TCP 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 obtener más información sobre la API, consulte [AuthorizeSecurityGroupEgress](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: este ejemplo define una regla de salida para el grupo de seguridad especificado para EC2-VPC. La regla concede acceso al rango de direcciones IP especificado en el puerto TCP 80. La sintaxis utilizada en este ejemplo requiere PowerShell la versión 3 o superior.

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

Ejemplo 2: Con la PowerShell versión 2, debe usar `New-Object` para crear el `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
```

Ejemplo 3: Este ejemplo concede acceso al grupo de seguridad de origen especificado en el puerto TCP 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 obtener más información sobre la API, consulte [AuthorizeSecurityGroupEgress](#) la referencia de AWS Tools for PowerShell cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo `AuthorizeSecurityGroupIngress` con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `AuthorizeSecurityGroupIngress`.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en contexto en el siguiente ejemplo de código:

- [Comience a utilizar instancias](#)

### .NET

#### AWS SDK for .NET

#### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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)
{
    // Get the IP address for the local computer.
    var ipAddress = await GetIpAddress();
    Console.WriteLine($"Your IP address is: {ipAddress}");
    var ipRanges = new List<IpRange> { new IpRange { CidrIp =
    $"{ipAddress}/32" } };
    var permission = new IpPermission
    {
        Ipv4Ranges = ipRanges,
        IpProtocol = "tcp",
        FromPort = 22,
        ToPort = 22
    };
    var permissions = new List<IpPermission> { permission };
```

```

        var response = await _amazonEC2.AuthorizeSecurityGroupIngressAsync(
            new AuthorizeSecurityGroupIngressRequest(groupName, permissions));
        return response.HttpStatusCode == HttpStatusCode.OK;
    }

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

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

```

- Para obtener más información sobre la API, consulta [AuthorizeSecurityGroupIngress](#) la Referencia AWS SDK for .NET de la API.

## Bash

### AWS CLI con el script Bash

#### Note

Hay más información. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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 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

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
}

```

Las funciones de utilidad utilizadas en este ejemplo.

```

#####
# 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 obtener más información sobre la API, consulta [AuthorizeSecurityGroupIngress](#) la Referencia de AWS CLI comandos.

## C++

### SDK para C++

#### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

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

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

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

authorize_request.AddIpPermissions(permission1);

Aws::EC2::Model::IpPermission permission2;
permission2.SetIpProtocol("tcp");
permission2.SetToPort(22);
```

```

permission2.SetFromPort(22);
permission2.AddIpRanges(ip_range);

authorize_request.AddIpPermissions(permission2);

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

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

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

```

- Para obtener más información sobre la API, consulta [AuthorizeSecurityGroupIngress](#) la Referencia AWS SDK for C++ de la API.

## CLI

### AWS CLI

Ejemplo 1: Agregar una regla que permita el tráfico SSH entrante

En el siguiente ejemplo de `authorize-security-group-ingress`, se agrega una regla que permite el tráfico entrante en un puerto TCP 22 (SSH).

```

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

```

Salida:

```

{
  "Return": true,

```

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

Ejemplo 2: Agregar una regla que permita el tráfico HTTP entrante desde otro grupo de seguridad

En el siguiente ejemplo de `authorize-security-group-ingress`, se agrega una regla que permite el acceso entrante al puerto TCP 80 desde el grupo de seguridad de origen `sg-1a2b3c4d`. El grupo de origen debe estar en la misma VPC o en una VPC del mismo nivel (requiere una conexión de emparejamiento de VPC). Se permite el tráfico entrante según las direcciones IP privadas de las instancias asociadas al grupo de seguridad de origen (y no la dirección IP pública o la dirección IP elástica).

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

Salida:

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

```

        "ToPort": 80,
        "ReferencedGroupInfo": {
            "GroupId": "sg-1a2b3c4d",
            "UserId": "123456789012"
        }
    }
]
}

```

### Ejemplo 3: Agregar varias reglas en la misma llamada

En el siguiente ejemplo de `authorize-security-group-ingress`, se usa el parámetro `ip-permissions` para agregar dos reglas de entrada, una que habilita el acceso entrante en el puerto TCP 3389 (RDP) y la otra 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, IpRanges =
"ToPort[={=172.31.0.0/16}]" CidrIp IpProtocol FromPort ToPort IpRanges CidrIp
```

Salida:

```

{
  "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"
    }
  ]
}

```

```

    }
  ]
}

```

#### Ejemplo 4: Agregar una regla para el tráfico ICMP

En el siguiente ejemplo de `authorize-security-group-ingress`, se usa el parámetro `ip-permissions` para agregar una regla de entrada que permita enviar el mensaje ICMP Destination Unreachable: Fragmentation Needed and Don't Fragment was Set (tipo 3, código 4) desde cualquier lugar.

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

Salida:

```

{
  "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"
    }
  ]
}

```

#### Ejemplo 5: Agregar una regla para el tráfico IPv6

En el siguiente ejemplo de `authorize-security-group-ingress`, se usa el parámetro `ip-permissions` para agregar una regla de entrada que permita el acceso SSH (puerto 22) desde el intervalo IPv6 `2001:db8:1234:1a00::/64`.

```
aws ec2 authorize-security-group-ingress --group-id sg-1234567890abcdef0 --ip-permissions
IpProtocol =tcp, =22, FromPort ToPort =22, Ipv6Ranges= "[{CidrIpv6=2001:db 8:1234:1
a00: :/64}]»
```

**Salida:**

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

**Ejemplo 6: Agregar una regla para el tráfico ICMPv6**

En el siguiente ejemplo de `authorize-security-group-ingress`, se usa el parámetro `ip-permissions` para agregar una regla de entrada que permita el tráfico ICMPv6 desde cualquier lugar.

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

**Salida:**

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



```
}
```

### Ejemplo 7: Agregar una regla con una descripción

En el siguiente ejemplo de `authorize-security-group-ingress`, se usa el parámetro `ip-permissions` para agregar una regla de entrada que permita el tráfico RDP desde el intervalo de direcciones IPv4 especificado. La regla incluye una descripción que lo ayudará a identificarla posteriormente.

```
aws ec2 authorize-security-group-ingress --group-id sg-1234567890abcdef0 --ip-  
permissions IpProtocol=tcp,=3389,FromPort=3389,IpRanges=["{{CidrIp=203.0.113.0/24,  
description='Acceso RDP desde una oficina de Nueva York'}}] --ToPort
```

Salida:

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

### Ejemplo 8: Agregar una regla de entrada que use una lista de prefijos

En el siguiente ejemplo de `authorize-security-group-ingress`, se usa el parámetro `ip-permissions` para agregar una regla de entrada que permita todo el tráfico de los rangos de CIDR de la lista de prefijos especificada.

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

Salida:

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

Para obtener más información, consulte [Grupo de seguridad](#) en la Guía del usuario de Amazon VPC.

- Para obtener más información sobre la API, consulte la Referencia de [AuthorizeSecurityGroupIngress](#) comandos AWS CLI .

## Java

### SDK para Java 2.x

#### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

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

```
        .build());

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

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

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

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

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

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

- Para obtener más información sobre la API, consulta [AuthorizeSecurityGroupIngress](#) la Referencia AWS SDK for Java 2.x de la API.

## JavaScript

### SDK para JavaScript (v3)

#### Note

Hay más información. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

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

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

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

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

- Para obtener más información sobre la API, consulta [AuthorizeSecurityGroupIngress](#) la Referencia AWS SDK for JavaScript de la API.

## Kotlin

### SDK para Kotlin

#### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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 obtener más información sobre la API, consulta [AuthorizeSecurityGroupIngress](#) la referencia sobre el AWS SDK para la API de Kotlin.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: en este ejemplo se definen las reglas de entrada de un grupo de seguridad para EC2-VPC. Estas reglas otorgan acceso a una dirección IP específica para SSH (puerto 22) y RDC (puerto 3389). Tenga en cuenta que debe identificar los grupos de seguridad para EC2-VPC mediante el ID del grupo de seguridad, no el nombre del grupo de seguridad. La sintaxis utilizada en este ejemplo requiere la PowerShell versión 3 o 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 )
```

Ejemplo 2: Con la PowerShell versión 2, debe usar `New-Object` para crear los `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 )
```

Ejemplo 3: En este ejemplo se definen las reglas de entrada de un grupo de seguridad de EC2-Classic. Estas reglas otorgan acceso a una dirección IP específica para SSH (puerto 22) y RDC (puerto 3389). La sintaxis utilizada en este ejemplo requiere la PowerShell versión 3 o 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 )
```

Ejemplo 4: Con la PowerShell versión 2, debe usar `New-Object` para crear los `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 )
```

Ejemplo 5: Este ejemplo concede al puerto TCP 8081 acceso desde el grupo de seguridad de origen especificado (sg-1a2b3c4d) al grupo de seguridad 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 } )
```

Ejemplo 6: Este ejemplo agrega el CIDR 5.5.5.5/32 a las reglas de ingreso del grupo de seguridad sg-1234abcd para el tráfico del puerto TCP 22 con una descripción.

```
$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 obtener [AuthorizeSecurityGroupIngress](#) más información sobre la API, consulte la referencia de cmdlets.AWS Tools for PowerShell



## Python

### SDK para Python (Boto3)

#### Note

Hay más información al respecto. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

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

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

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

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

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

```

```

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

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

```

- Para obtener más información sobre la API, consulta [AuthorizeSecurityGroupIngress](#) la AWS Referencia de API de SDK for Python (Boto3).

Para obtener una lista completa de las guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **CancelCapacityReservation** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `CancelCapacityReservation`.

## CLI

### AWS CLI

Para cancelar una reserva de capacidad

El siguiente `cancel-capacity-reservation` ejemplo cancela la reserva de capacidad especificada.

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

Salida:

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

Para obtener más información, consulte [Cancelar una reserva de capacidad](#) en la Guía del usuario de Amazon Elastic Compute Cloud para instancias de Linux.

- Para obtener más información sobre la API, consulte [CancelCapacityReservation](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: En este ejemplo se cancela la reserva de capacidad `cr-0c1f2345db6f7cdba`

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

Salida:

```
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 obtener más información [CancelCapacityReservations](#) sobre AWS Tools for PowerShell la API, consulte la referencia del cmdlet.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **CancelImportTask** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `CancelImportTask`.

### CLI

#### AWS CLI

Para cancelar una tarea de importación

El siguiente `cancel-import-task` ejemplo cancela la tarea de importación de imágenes especificada.

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

Salida:

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

- Para obtener más información sobre la API, consulte [CancelImportTask](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: En este ejemplo se cancela la tarea de importación especificada (importación de instantáneas o imágenes). Si es necesario, se puede proporcionar un motivo mediante el **-CancelReason** parámetro.

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

- Para obtener más información sobre la API, consulte [CancelImportTask](#) la referencia del AWS Tools for PowerShell cmdlet.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **CancelSpotFleetRequests** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `CancelSpotFleetRequests`.

### CLI

#### AWS CLI

Ejemplo 1: Cancelar una solicitud de flota de Spot y cancelar las instancias asociadas

En el siguiente `cancel-spot-fleet-requests` ejemplo, se cancela una solicitud de flota puntual y se cancelan las instancias puntuales y bajo demanda asociadas.

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

Salida:

```
{  
  "SuccessfulFleetRequests": [  
    {
```

```

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

```

Para obtener más información, consulte [Cancelar una solicitud de flota puntual](#) en la Guía del usuario de Amazon Elastic Compute Cloud para instancias de Linux.

Ejemplo 2: Para cancelar una solicitud de flota puntual sin cancelar las instancias asociadas

En el siguiente `cancel-spot-fleet-requests` ejemplo, se cancela una solicitud de flota puntual sin cancelar las instancias puntuales y bajo demanda asociadas.

```

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

```

Salida:

```

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

```

Para obtener más información, consulte [Cancelar una solicitud de flota puntual](#) en la Guía del usuario de Amazon Elastic Compute Cloud para instancias de Linux.

- Para obtener más información sobre la API, consulte [CancelSpotFleetRequests](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: En este ejemplo se cancela la solicitud de flota de Spot especificada y se cancelan las instancias de Spot asociadas.

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

Ejemplo 2: Este ejemplo cancela la solicitud de flota de Spot especificada sin cancelar las instancias de Spot asociadas.

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

- Para obtener más información sobre la API, consulte la referencia del [CancelSpotFleetRequests AWS Tools for PowerShell](#) cmdlet.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **CancelSpotInstanceRequests** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `CancelSpotInstanceRequests`.

### CLI

#### AWS CLI

Para cancelar las solicitudes de instancias puntuales

Este comando de ejemplo cancela una solicitud de instancia puntual.

Comando:

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

Salida:

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

- Para obtener más información sobre la API, consulte [CancelSpotInstanceRequests](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: En este ejemplo se cancela la solicitud de instancia de spot especificada.

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

Salida:

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

- Para obtener información sobre la API, consulte [CancelSpotInstanceRequests AWS Tools for PowerShell](#) Cmdlet Reference.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **ConfirmProductInstance** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `ConfirmProductInstance`.



## CLI

### AWS CLI

Para confirmar la instancia del producto

En este ejemplo se determina si el código de producto especificado está asociado a la instancia especificada.

Comando:

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

Salida:

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

- Para obtener más información sobre la API, consulte [ConfirmProductInstance](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: Este ejemplo determina si el código de producto especificado está asociado a la instancia especificada.

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

- Para obtener más información sobre la API, consulte [ConfirmProductInstance](#) la referencia del AWS Tools for PowerShell cmdlet.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **CopyImage** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar CopyImage.

### CLI

#### AWS CLI

Ejemplo 1: Para copiar una AMI a otra región

El siguiente comando de `copy-image` ejemplo copia la AMI especificada de la `us-west-2` región a la `us-east-1` región y agrega una breve descripción.

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

Salida:

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

Para obtener más información, consulte [Copiar una AMI](#) en la Guía del usuario de Amazon EC2.

Ejemplo 2: Para copiar una AMI a otra región y cifrar la instantánea de respaldo

El siguiente `copy-image` comando copia la AMI especificada de la `us-west-2` región a la región actual y cifra la instantánea de respaldo mediante la clave KMS especificada.

```
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
```

**Salida:**

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

Para obtener más información, consulte [Copiar una AMI](#) en la Guía del usuario de Amazon EC2.

Ejemplo 3: Para incluir las etiquetas de AMI definidas por el usuario al copiar una AMI

El `copy-image` comando siguiente utiliza el `--copy-image-tags` parámetro para copiar las etiquetas de AMI definidas por el usuario al copiar la AMI.

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

**Salida:**

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

Para obtener más información, consulte [Copiar una AMI](#) en la Guía del usuario de Amazon EC2.

- Para obtener más información sobre la API, consulte [CopyImage](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: En este ejemplo, se copia la AMI especificada de la región «UE (Irlanda)» a la región «EE.UU. Oeste (Oregón)». Si no se especifica `-Region`, la región predeterminada actual se utiliza como región de destino.

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

Salida:

```
ami-87654321
```

- Para obtener más información sobre la API, consulte la referencia [CopyImage](#) de AWS Tools for PowerShell cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **CopySnapshot** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar CopySnapshot.

CLI

### AWS CLI

Ejemplo 1: Para copiar una instantánea a otra región

El siguiente comando de `copy-snapshot` ejemplo copia la instantánea especificada de la `us-west-2` región a la `us-east-1` región y añade una breve descripción.

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

Salida:

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

Para obtener más información, consulte [Copiar una instantánea de Amazon EBS](#) en la Guía del usuario de Amazon EC2.

Ejemplo 2: Para copiar una instantánea no cifrada y cifrar la nueva instantánea

El siguiente `copy-snapshot` comando copia la instantánea no cifrada especificada de la `us-west-2` región a la región actual y cifra la nueva instantánea con la clave KMS especificada.

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

Salida:

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

Para obtener más información, consulte [Copiar una instantánea de Amazon EBS](#) en la Guía del usuario de Amazon EC2.

- Para obtener más información sobre la API, consulte la Referencia [CopySnapshot](#) de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: En este ejemplo, se copia la instantánea especificada de la región de la UE (Irlanda) a la región de EE.UU. Oeste (Oregón).

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

Ejemplo 2: Si establece una región predeterminada y omite el parámetro Región, la región de destino predeterminada será la región predeterminada.

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

- Para obtener más información sobre la API, consulte la referencia [CopySnapshot](#) de AWS Tools for PowerShell cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **CreateCapacityReservation** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `CreateCapacityReservation`.

### CLI

#### AWS CLI

Ejemplo 1: Para crear una reserva de capacidad

El siguiente `create-capacity-reservation` ejemplo crea una reserva de capacidad en la zona de `eu-west-1a` disponibilidad, en la que puede lanzar tres `t2.medium` instancias que ejecuten un sistema operativo Linux/Unix. De forma predeterminada, la reserva de capacidad se crea con criterios de coincidencia de instancias abiertas y no admite el almacenamiento efímero, y permanece activa hasta que la canceles manualmente.

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

Salida:

```
{  
  "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"
  }
}

```

**Ejemplo 2:** Para crear una reserva de capacidad que finalice automáticamente en una fecha y hora especificadas

El siguiente `create-capacity-reservation` ejemplo crea una reserva de capacidad en la zona de `eu-west-1a` disponibilidad, en la que puede lanzar tres `m5.large` instancias que ejecuten un sistema operativo Linux/Unix. Esta reserva de capacidad finaliza automáticamente el 31 de agosto de 2019 a las 23:59:59.

```

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

```

**Salida:**

```

{
  "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"
  }
}
```

Ejemplo 3: Para crear una reserva de capacidad que solo acepte lanzamientos de instancias segmentadas

En el siguiente `create-capacity-reservation` ejemplo, se crea una reserva de capacidad que solo acepta los lanzamientos de instancias de destino.

```
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
```

Salida:

```
{
  "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 obtener más información, consulte [Crear una reserva de capacidad](#) en la Guía del usuario de Amazon Elastic Compute Cloud para instancias de Linux.

- Para obtener más información sobre la API, consulte [CreateCapacityReservation](#) la Referencia de AWS CLI comandos.



## PowerShell

### Herramientas para PowerShell

Ejemplo 1: En este ejemplo se crea una nueva reserva de capacidad con los atributos especificados

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

Salida:

```
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 obtener más información sobre la API, consulte [CreateCapacityReservation](#) la referencia del AWS Tools for PowerShell cmdlet.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **CreateCustomerGateway** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar CreateCustomerGateway.

## CLI

### AWS CLI

Para crear una pasarela de clientes

En este ejemplo, se crea una puerta de enlace para el cliente con la dirección IP especificada para su interfaz exterior.

Comando:

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

Salida:

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

- Para obtener más información sobre la API, consulte [CreateCustomerGateway](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: En este ejemplo se crea la pasarela de clientes especificada.

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

Salida:

```
BgpAsn          : 65534
```

```
CustomerGatewayId : cgw-1a2b3c4d
IpAddress          : 203.0.113.12
State              : available
Tags               : {}
Type               : ipsec.1
```

- Para obtener más información sobre la API, consulte [CreateCustomerGateway](#) la referencia del AWS Tools for PowerShell cmdlet.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **CreateDhcpOptions** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `CreateDhcpOptions`.

### CLI

#### AWS CLI

Para crear un conjunto de opciones de DHCP

El siguiente `create-dhcp-options` ejemplo crea un conjunto de opciones de DHCP que especifica el nombre de dominio, los servidores de nombres de dominio y el tipo de nodo NetBIOS.

```
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"
```

Salida:

```
{
  "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 obtener más información sobre la API, consulte la Referencia [CreateDhcpOptions](#) de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: En este ejemplo se crea el conjunto especificado de opciones de DHCP. La sintaxis utilizada en este ejemplo requiere la PowerShell versión 3 o 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
```

**Salida:**

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

Ejemplo 2: Con la PowerShell versión 2, debe usar `New-Object` para crear cada opción de DHCP.

```
$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)
```

**Salida:**

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

- Para obtener más información sobre la API, consulte la referencia de [CreateDhcpOptions](#) cmdlets AWS Tools for PowerShell .

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **CreateFlowLogs** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `CreateFlowLogs`.

**CLI****AWS CLI**

Ejemplo 1: Para crear un registro de flujo

El siguiente `create-flow-logs` ejemplo crea un registro de flujo que captura todo el tráfico rechazado para la interfaz de red especificada. Los registros de flujo se envían a un grupo de CloudWatch registros en Logs mediante los permisos de la función de IAM 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
```

Salida:

```
{  
  "ClientToken": "so0eNA2uSHUN1HI0S2cJ305GuIX1CezaRdGtexample",  
  "FlowLogIds": [  
    "fl-12345678901234567"  
  ],  
  "Unsuccessful": []  
}
```

Para obtener más información, consulte [Logs de flujo de VPC](#) en la Guía del usuario de Amazon VPC.

Ejemplo 2: Para crear un registro de flujo con un formato personalizado

El siguiente `create-flow-logs` ejemplo crea un registro de flujo que captura todo el tráfico de la VPC especificada y entrega los registros de flujo a un bucket de Amazon S3. El parámetro `--log-format` especifica un formato personalizado para las entradas de registros de flujo. Para ejecutar este comando en Windows, cambie las comillas simples (') por comillas dobles («).

```
aws ec2 create-flow-logs \  
  --resource-type VPC \  
  --resource-ids vpc-001122333444556677 \  
  --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 obtener más información, consulte [Logs de flujo de VPC](#) en la Guía del usuario de Amazon VPC.

Ejemplo 3: Para crear un registro de flujo con un intervalo de agregación máximo de un minuto

El siguiente `create-flow-logs` ejemplo crea un registro de flujo que captura todo el tráfico de la VPC especificada y entrega los registros de flujo a un bucket de Amazon S3. El `--max-aggregation-interval` parámetro especifica un intervalo de agregación máximo 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 obtener más información, consulte [Logs de flujo de VPC](#) en la Guía del usuario de Amazon VPC.

- Para obtener más información sobre la API, consulte [CreateFlowLogs](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: En este ejemplo se crea un registro de flujo de EC2 para la subred-1d234567 en la `cloud-watch-log` denominada «subnet1-log» para todo el tráfico de «REJECT» utilizando los permisos de la función de «Administrador»

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

Salida:

ClientToken	FlowLogIds	Unsuccessful
-----	-----	-----

```
m1VN2cxP3iB4qo//VUK15EU6cF7gQL0xcqNefvjeTGw= {f1-012fc34eed5678c9d} {}
```

- Para obtener AWS Tools for PowerShell más información sobre la API, consulte la referencia de cmdlets. [CreateFlowLogs](#)

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **CreateImage** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar CreateImage.

### CLI

#### AWS CLI

Ejemplo 1: Para crear una AMI a partir de una instancia respaldada por Amazon EBS

En el siguiente `create-image` ejemplo, se crea una AMI a partir de la instancia especificada.

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

Salida:

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

Para obtener más información sobre cómo especificar un mapeo de dispositivos de bloques para su AMI, consulte [Especificar un mapeo de dispositivos de bloques para una AMI](#) en la Guía del usuario de Amazon EC2.

Ejemplo 2: Para crear una AMI a partir de una instancia respaldada por Amazon EBS sin reiniciar

En el siguiente `create-image` ejemplo, se crea una AMI y se establece el parámetro `--no-reboot` para que la instancia no se reinicie antes de crear la imagen.



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

Salida:

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

Para obtener más información sobre cómo especificar un mapeo de dispositivos de bloques para su AMI, consulte [Especificar un mapeo de dispositivos de bloques para una AMI](#) en la Guía del usuario de Amazon EC2.

Ejemplo 3: Para etiquetar una AMI y las instantáneas al crearlas

El siguiente `create-image` ejemplo crea una AMI y etiqueta la AMI y las instantáneas con la misma etiqueta. `cost-center=cc123`

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

Salida:

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

Para obtener más información sobre cómo etiquetar los recursos al crearlos, consulte [Añadir etiquetas al crear los recursos en la](#) Guía del usuario de Amazon EC2.

- Para obtener más información sobre la API, consulte la Referencia [CreateImage](#) de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: en este ejemplo se crea una AMI con el nombre y la descripción especificados, a partir de la instancia especificada. Amazon EC2 intenta cerrar la instancia de forma limpia antes de crear la imagen y la reinicia al finalizar.

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

Ejemplo 2: en este ejemplo se crea una AMI con el nombre y la descripción especificados, a partir de la instancia especificada. Amazon EC2 crea la imagen sin cerrar ni reiniciar la instancia; por lo tanto, no se puede garantizar la integridad del sistema de archivos de la imagen creada.

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

Ejemplo 3: En este ejemplo se crea una AMI con tres volúmenes. El primer volumen se basa en una instantánea de Amazon EBS. El segundo volumen es un volumen Amazon EBS vacío de 100 GiB. El tercer volumen es un volumen de almacén de instancias. La sintaxis utilizada en este ejemplo requiere PowerShell la versión 3 o 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 obtener más información sobre la API, consulte [CreateImage](#) la referencia de AWS Tools for PowerShell cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo `CreateInstanceExportTask` con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `CreateInstanceExportTask`.

### CLI

#### AWS CLI

Para exportar una instancia

Este comando de ejemplo crea una tarea para exportar la instancia `i-1234567890abcdef0` al bucket `myexportbucket` de 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
```

Salida:

```
{
  "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"
  }
}
```

- AWS CLI Para obtener [CreateInstanceExportTask](#) más información sobre la API, consulte la Referencia de comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: En este ejemplo, se exporta una instancia detenida **i-0800b00a00EXAMPLE**, como un disco duro virtual (VHD) al bucket **testbucket-export-instances-2019** de S3. El entorno de destino es **Microsoft**, y el parámetro **region** se añade porque la instancia está en la **us-east-1** región, mientras que la AWS región predeterminada del usuario no es **us-east-1**. Para obtener el estado de la tarea de exportación, copia el **ExportTaskId** valor de los resultados de este comando y ejecuta **Get-EC2ExportTask -ExportTaskId export\_task\_ID\_from\_results**.

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

### Salida:

```
Description           :
ExportTaskId          : export-i-077c73108aEXAMPLE
ExportToS3Task        : Amazon.EC2.Model.ExportToS3Task
InstanceExportDetails : Amazon.EC2.Model.InstanceExportDetails
State                  : active
StatusMessage         :
```

- Para obtener más información sobre la API, consulte [CreateInstanceExportTask](#) la referencia de AWS Tools for PowerShell cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **CreateInternetGateway** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar **CreateInternetGateway**.

## CLI

### AWS CLI

Para crear una puerta de enlace a Internet

En el siguiente `create-internet-gateway` ejemplo, se crea una puerta de enlace a Internet con la etiqueta `Name=my-igw`.

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

Salida:

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

Para obtener más información, consulte [Puertas de enlace de Internet](#) en la Guía del usuario de Amazon VPC.

- Para obtener más información sobre la API, consulte [CreateInternetGateway](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: En este ejemplo se crea una puerta de enlace a Internet.

```
New-EC2InternetGateway
```

**Salida:**

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

- Para obtener más información sobre la API, consulte [CreateInternetGateway](#) la referencia del AWS Tools for PowerShell cmdlet.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **CreateKeyPair** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `CreateKeyPair`.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en contexto en el siguiente ejemplo de código:

- [Comience a utilizar instancias](#)

### .NET

#### AWS SDK for .NET

#### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

```

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

```

```
var request = new CreateKeyPairRequest
{
    KeyName = keyPairName,
};

var response = await _amazonEC2.CreateKeyPairAsync(request);

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

/// <summary>
/// Save KeyPair information to a temporary file.
/// </summary>
/// <param name="keyPair">The name of the key pair.</param>
/// <returns>The full path to the temporary file.</returns>
public string SaveKeyPair(KeyPair keyPair)
{
    var tempPath = Path.GetTempPath();
    var tempFileName = $"{tempPath}\\{Path.GetRandomFileName()}";
    var pemFileName = Path.ChangeExtension(tempFileName, "pem");

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

    return pemFileName;
}
```

- Para obtener más información sobre la API, consulta [CreateKeyPair](#) la Referencia AWS SDK for .NET de la API.

## Bash

## AWS CLI con el script Bash

 Note

Hay más información. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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
}
```

Las funciones de utilidad utilizadas en este ejemplo.

```
#####
# 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 obtener más información sobre la API, consulta [CreateKeyPair](#) la Referencia de AWS CLI comandos.

## C++

### SDK para C++

#### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

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

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

- Para obtener más información sobre la API, consulta [CreateKeyPair](#) la Referencia AWS SDK for C++ de la API.

## CLI

### AWS CLI

#### Crear un par de claves

En este ejemplo, se crea un par de claves denominado MyKeyPair.

Comando:

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

El resultado es una versión ASCII de la clave privada y la huella digital de la clave. Debe guardar la clave en un archivo.

Para obtener más información, consulte [Uso del par de claves](#) en la Guía del usuario de la Interfaz de la línea de comandos de AWS .

- Para obtener más información sobre la API, consulta [CreateKeyPair](#) la Referencia de AWS CLI comandos.

## Java

### SDK para Java 2.x

#### Note

Hay más información al respecto [en GitHub](#). Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

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

        CreateKeyPairResponse response = ec2.createKeyPair(request);
        String content = response.keyMaterial();
```

```
        BufferedWriter writer = new BufferedWriter(new FileWriter(fileName));
        writer.write(content);
        writer.close();
        System.out.println("Successfully created key pair named " + keyName);

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

- Para obtener más información sobre la API, consulta [CreateKeyPair](#) la Referencia AWS SDK for Java 2.x de la API.

## JavaScript

### SDK para JavaScript (v3)

#### Note

Hay más información. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

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

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

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

```
}  
};
```

- Para obtener más información sobre la API, consulta [CreateKeyPair](#) la Referencia AWS SDK for JavaScript de la API.

## Kotlin

### SDK para Kotlin

#### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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 obtener más información sobre la API, consulta [CreateKeyPair](#) la referencia sobre el AWS SDK para la API de Kotlin.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: Este ejemplo crea un key pair y captura la clave privada RSA codificada en PEM en un archivo con el nombre especificado. Cuando la utilice PowerShell, la codificación debe estar establecida en ascii para generar una clave válida. Para obtener más

información, consulte [Crear, mostrar y eliminar pares de claves de Amazon EC2 \(https://docs.aws.amazon.com/cli/latest/userguide/cli-services-ec2-keypairs.html\)](https://docs.aws.amazon.com/cli/latest/userguide/cli-services-ec2-keypairs.html) en la Guía del usuario de la interfaz de línea de AWS comandos.

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

- Para obtener más información sobre la API, consulte la referencia [CreateKeyPair](#) de AWS Tools for PowerShell cmdlets.

## Python

### SDK para Python (Boto3)

#### Note

Hay más información al respecto. [GitHub](#) Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

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

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

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

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

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

- Para obtener más información sobre la API, consulta [CreateKeyPair](#) la AWS Referencia de API de SDK for Python (Boto3).



## Ruby

### SDK para Ruby

#### Note

Hay más información al respecto. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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}'."
  return true
rescue Aws::EC2::Errors::InvalidKeyPairDuplicate
  puts "Error creating key pair: a key pair named '#{key_pair_name}' " \
    "already exists."
  return false
rescue StandardError => e
  puts "Error creating key pair or saving private key file: #{e.message}"
```

```
    return false
  end

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

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

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

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

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

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

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

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

```

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

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

run_me if $PROGRAM_NAME == __FILE__

```

- Para obtener más información sobre la API, consulta [CreateKeyPair](#) la Referencia AWS SDK for Ruby de la API.

## SAP ABAP

### SDK de SAP ABAP

#### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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 obtener más información sobre la API, consulte [CreateKeyPair](#) la referencia sobre la API ABAP del AWS SDK para SAP.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte [Cree recursos de Amazon EC2 mediante un SDK AWS](#). En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **CreateLaunchTemplate** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `CreateLaunchTemplate`.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en contexto en el siguiente ejemplo de código:

- [Creación y administración de un servicio resiliente](#)

.NET

AWS SDK for .NET

### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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)
    {
        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;
    }
}

```

- Para obtener más información sobre la API, consulta [CreateLaunchTemplate](#) la Referencia AWS SDK for .NET de la API.

## CLI

## AWS CLI

## Ejemplo 1: Crear una plantilla de lanzamiento

En el siguiente ejemplo de `create-launch-template`, se crea una plantilla de lanzamiento que especifica la subred en la que se lanza la instancia, se asigna una dirección IP pública y una dirección IPv6 a la instancia y se crea una etiqueta para la instancia.

```
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"}]}]}'
```

## Salida:

```
{  
  "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 obtener más información, consulte [Lanzamiento de una instancia desde una plantilla de lanzamiento](#) en la Guía del usuario de Amazon Elastic Compute Cloud. Para obtener información acerca de cómo citar parámetros con formato JSON, consulte [Entrecomillado de cadenas](#) en la Guía del usuario de la Interfaz de la línea de comandos de AWS .

## Ejemplo 2: Crear una plantilla de lanzamiento para Amazon EC2 Auto Scaling

En el siguiente ejemplo de `create-launch-template`, se crea una plantilla de lanzamiento con varias etiquetas y una asignación de dispositivos de bloques para especificar un volumen de EBS adicional cuando se lanza una instancia. Especifique un valor para `Groups` que

corresponda a los grupos de seguridad de la VPC en la que el grupo de escalado automático lanzará las instancias. Especifique las subredes de la VPC como propiedades del grupo de escalado automático.

```
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
```

Salida:

```
{
  "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 obtener más información, consulte [Creación de una plantilla de lanzamiento para un grupo de escalado automático](#) en la Guía del usuario de Amazon EC2 Auto Scaling. Para obtener información acerca de cómo citar parámetros con formato JSON, consulte [Entrecorillado de cadenas](#) en la Guía del usuario de la Interfaz de la línea de comandos de AWS .

**Ejemplo 3:** Crear una plantilla de lanzamiento que especifique el cifrado de los volúmenes de EBS

En el siguiente ejemplo de `create-launch-template`, se crea una plantilla de lanzamiento que incluye volúmenes de EBS cifrados creados a partir de una instantánea no cifrada.



También etiqueta los volúmenes durante la creación. Si el cifrado está deshabilitado de forma predeterminada, debe especificar la opción "Encrypted" que se muestra en el siguiente ejemplo. Si usa la opción "KmsKeyId" para especificar una CMK administrada por el cliente, también debe especificar la opción "Encrypted" aunque el cifrado esté habilitado de forma predeterminada.

```
aws ec2 create-launch-template \  
  --launch-template-name TemplateForEncryption \  
  --launch-template-data file://config.json
```

Contenidos 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"  
        }  
      ]  
    }  
  ]  
}
```

Salida:

```
{
  "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 obtener más información, consulte Restauración de un volumen de Amazon EBS a partir de una instantánea en la Guía del usuario de Amazon Elastic Compute Cloud.

- Para obtener más información sobre la API, consulta [CreateLaunchTemplate](#) la Referencia de AWS CLI comandos.

## JavaScript

### SDK para JavaScript (v3)

#### Note

Hay más información. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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 obtener más información sobre la API, consulta [CreateLaunchTemplate](#) la Referencia AWS SDK for JavaScript de la API.

## Python

### SDK para Python (Boto3)

#### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

En este ejemplo, se crea una plantilla de lanzamiento que incluye un perfil de instancia que concede permisos específicos a la instancia y un script Bash de datos de usuario que se ejecuta en la instancia una vez iniciada.

```
class AutoScaler:  
    """  
    Encapsulates Amazon EC2 Auto Scaling and EC2 management actions.  
    """  
  
    def __init__(  
        self,  
        resource_prefix,  
        inst_type,  
        ami_param,  
        autoscaling_client,  
        ec2_client,  
        ssm_client,  
        iam_client,  
    ):  
        """
```

```

        :param resource_prefix: The prefix for naming AWS resources that are
        created by this class.
        :param inst_type: The type of EC2 instance to create, such as t3.micro.
        :param ami_param: The Systems Manager parameter used to look up the AMI
        that is
                created.
        :param autoscaling_client: A Boto3 EC2 Auto Scaling client.
        :param ec2_client: A Boto3 EC2 client.
        :param ssm_client: A Boto3 Systems Manager client.
        :param iam_client: A Boto3 IAM client.
        """
        self.inst_type = inst_type
        self.ami_param = ami_param
        self.autoscaling_client = autoscaling_client
        self.ec2_client = ec2_client
        self.ssm_client = ssm_client
        self.iam_client = iam_client
        self.launch_template_name = f"{resource_prefix}-template"
        self.group_name = f"{resource_prefix}-group"
        self.instance_policy_name = f"{resource_prefix}-pol"
        self.instance_role_name = f"{resource_prefix}-role"
        self.instance_profile_name = f"{resource_prefix}-prof"
        self.bad_creds_policy_name = f"{resource_prefix}-bc-pol"
        self.bad_creds_role_name = f"{resource_prefix}-bc-role"
        self.bad_creds_profile_name = f"{resource_prefix}-bc-prof"
        self.key_pair_name = f"{resource_prefix}-key-pair"

    def create_template(self, server_startup_script_file, instance_policy_file):
        """
        Creates an Amazon EC2 launch template to use with Amazon EC2 Auto
        Scaling. The
        launch template specifies a Bash script in its user data field that runs
        after
        the instance is started. This script installs Python packages and starts
        a
        Python web server on the instance.

        :param server_startup_script_file: The path to a Bash script file that is
        run
                when an instance starts.
        :param instance_policy_file: The path to a file that defines a
        permissions policy

```

```

                                to create and attach to the instance
profile.
:return: Information about the newly created template.
"""
template = {}
try:
    self.create_key_pair(self.key_pair_name)
    self.create_instance_profile(
        instance_policy_file,
        self.instance_policy_name,
        self.instance_role_name,
        self.instance_profile_name,
    )
    with open(server_startup_script_file) as file:
        start_server_script = file.read()
    ami_latest = self.ssm_client.get_parameter(Name=self.ami_param)
    ami_id = ami_latest["Parameter"]["Value"]
    lt_response = self.ec2_client.create_launch_template(
        LaunchTemplateName=self.launch_template_name,
        LaunchTemplateData={
            "InstanceType": self.inst_type,
            "ImageId": ami_id,
            "IamInstanceProfile": {"Name": self.instance_profile_name},
            "UserData": base64.b64encode(
                start_server_script.encode(encoding="utf-8")
            ).decode(encoding="utf-8"),
            "KeyName": self.key_pair_name,
        },
    )
    template = lt_response["LaunchTemplate"]
    log.info(
        "Created launch template %s for AMI %s on %s.",
        self.launch_template_name,
        ami_id,
        self.inst_type,
    )
except ClientError as err:
    if (
        err.response["Error"]["Code"]
        == "InvalidLaunchTemplateName.AlreadyExistsException"
    ):
        log.info(
            "Launch template %s already exists, nothing to do.",
            self.launch_template_name,
```

```
        )
    else:
        raise AutoScalerError(
            f"Couldn't create launch template
            {self.launch_template_name}: {err}."
        )
    return template
```

- Para obtener más información sobre la API, consulta [CreateLaunchTemplate](#) la AWS Referencia de API de SDK for Python (Boto3).

Para obtener una lista completa de las guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **CreateNetworkAcl** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `CreateNetworkAcl`.

### CLI

#### AWS CLI

Para crear una ACL de red

En este ejemplo, se crea una ACL de red para la VPC especificada.

Comando:

```
aws ec2 create-network-acl --vpc-id vpc-a01106c2
```

Salida:

```
{
  "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 obtener más información sobre la API, consulte [CreateNetworkAcl](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: en este ejemplo se crea una ACL de red para la VPC especificada.

```
New-EC2NetworkAcl -VpcId vpc-12345678
```

### Salida:

```

Associations : {}
Entries      : {Amazon.EC2.Model.NetworkAclEntry,
               Amazon.EC2.Model.NetworkAclEntry}
IsDefault    : False
NetworkAclId : acl-12345678
Tags         : {}
VpcId        : vpc-12345678

```

- Para obtener información sobre la API, consulte la referencia [CreateNetworkAcl](#) de AWS Tools for PowerShell cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **CreateNetworkAclEntry** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `CreateNetworkAclEntry`.

### CLI

#### AWS CLI

Para crear una entrada de ACL de red

En este ejemplo, se crea una entrada para la ACL de red especificada. La regla permite la entrada de tráfico desde cualquier dirección IPv4 (0.0.0.0/0) del puerto UDP 53 (DNS) a cualquier subred asociada. Si el comando se ejecuta correctamente, no se muestra ningún resultado.

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
```

En este ejemplo, se crea una regla para la ACL de red especificada que permite la entrada de tráfico desde cualquier dirección IPv6 (:: /0) en el puerto TCP 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 obtener más información sobre la API, consulte la Referencia de [CreateNetworkAclEntry](#) comandos AWS CLI .



## PowerShell

### Herramientas para PowerShell

Ejemplo 1: En este ejemplo se crea una entrada para la ACL de red especificada. La regla permite el tráfico entrante desde cualquier lugar (0.0.0.0/0) del puerto UDP 53 (DNS) hacia cualquier subred asociada.

```
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 obtener más información sobre la API, consulte la referencia de cmdlets.  
[CreateNetworkAclEntry](#) AWS Tools for PowerShell

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **CreateNetworkInterface** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar CreateNetworkInterface.

### CLI

#### AWS CLI

Ejemplo 1: Para especificar una dirección IPv4 para una interfaz de red

El siguiente create-network-interface ejemplo crea una interfaz de red para la subred especificada con la dirección IPv4 principal especificada.

```
aws ec2 create-network-interface \  
  --subnet-id subnet-00a24d0d67acf6333 \  
  --description "my network interface" \  
  --groups sg-09dfba7ed20cda78b \  
  --private-ip-address 10.0.8.17
```

Salida:

```
{
  "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"
  }
}
```

Ejemplo 2: Para crear una interfaz de red con una dirección IPv4 y una dirección IPv6

El siguiente `create-network-interface` ejemplo crea una interfaz de red para la subred especificada con una dirección IPv4 y una dirección IPv6 seleccionadas por Amazon EC2.

```
aws ec2 create-network-interface \
  --subnet-id subnet-00a24d0d67acf6333 \
  --description "my dual stack network interface" \
  --ipv6-address-count 1 \
```

```
--groups sg-09dfba7ed20cda78b
```

**Salida:**

```
{
  "NetworkInterface": {
    "AvailabilityZone": "us-west-2a",
    "Description": "my dual stack network interface",
    "Groups": [
      {
        "GroupName": "my-security-group",
        "GroupId": "sg-09dfba7ed20cda78b"
      }
    ],
    "InterfaceType": "interface",
    "Ipv6Addresses": [
      {
        "Ipv6Address": "2600:1f13:cfe:3650:a1dc:237c:393a:4ba7",
        "IsPrimaryIpv6": false
      }
    ],
    "MacAddress": "06:b8:68:d2:b2:2d",
    "NetworkInterfaceId": "eni-05da417453f9a84bf",
    "OwnerId": "123456789012",
    "PrivateDnsName": "ip-10-0-8-18.us-west-2.compute.internal",
    "PrivateIpAddress": "10.0.8.18",
    "PrivateIpAddresses": [
      {
        "Primary": true,
        "PrivateDnsName": "ip-10-0-8-18.us-west-2.compute.internal",
        "PrivateIpAddress": "10.0.8.18"
      }
    ],
    "RequesterId": "AIDA4Z3Y7GSXTMEXAMPLE",
    "RequesterManaged": false,
    "SourceDestCheck": true,
    "Status": "pending",
    "SubnetId": "subnet-00a24d0d67acf6333",
    "TagSet": [],
    "VpcId": "vpc-02723a0feeeb9d57b",
    "Ipv6Address": "2600:1f13:cfe:3650:a1dc:237c:393a:4ba7"
  }
}
```

### Ejemplo 3: Para crear una interfaz de red con opciones de configuración de seguimiento de conexiones

El siguiente `create-network-interface` ejemplo crea una interfaz de red y configura los tiempos de espera del seguimiento de las conexiones inactivas.

```
aws ec2 create-network-interface \  
  --subnet-id subnet-00a24d0d67acf6333 \  
  --groups sg-02e57dbcfe0331c1b \  
  --connection-tracking-specification TcpEstablishedTimeout=86400,UdpTimeout=60
```

Salida:

```
{  
  "NetworkInterface": {  
    "AvailabilityZone": "us-west-2a",  
    "ConnectionTrackingConfiguration": {  
      "TcpEstablishedTimeout": 86400,  
      "UdpTimeout": 60  
    },  
    "Description": "",  
    "Groups": [  
      {  
        "GroupName": "my-security-group",  
        "GroupId": "sg-02e57dbcfe0331c1b"  
      }  
    ],  
    "InterfaceType": "interface",  
    "Ipv6Addresses": [],  
    "MacAddress": "06:4c:53:de:6d:91",  
    "NetworkInterfaceId": "eni-0c133586e08903d0b",  
    "OwnerId": "123456789012",  
    "PrivateDnsName": "ip-10-0-8-94.us-west-2.compute.internal",  
    "PrivateIpAddress": "10.0.8.94",  
    "PrivateIpAddresses": [  
      {  
        "Primary": true,  
        "PrivateDnsName": "ip-10-0-8-94.us-west-2.compute.internal",  
        "PrivateIpAddress": "10.0.8.94"  
      }  
    ],  
    "RequesterId": "AIDA4Z3Y7GSXTMEXAMPLE",  
    "RequesterManaged": false,  
  }  
}
```

```

    "SourceDestCheck": true,
    "Status": "pending",
    "SubnetId": "subnet-00a24d0d67acf6333",
    "TagSet": [],
    "VpcId": "vpc-02723a0feeb9d57b"
  }
}

```

#### Ejemplo 4: Para crear un adaptador Elastic Fabric

El siguiente `create-network-interface` ejemplo crea un EFA.

```

aws ec2 create-network-interface \
  --interface-type efa \
  --subnet-id subnet-00a24d0d67acf6333 \
  --description "my efa" \
  --groups sg-02e57dbcfe0331c1b

```

Salida:

```

{
  "NetworkInterface": {
    "AvailabilityZone": "us-west-2a",
    "Description": "my efa",
    "Groups": [
      {
        "GroupName": "my-efa-sg",
        "GroupId": "sg-02e57dbcfe0331c1b"
      }
    ],
    "InterfaceType": "efa",
    "Ipv6Addresses": [],
    "MacAddress": "06:d7:a4:f7:4d:57",
    "NetworkInterfaceId": "eni-034acc2885e862b65",
    "OwnerId": "123456789012",
    "PrivateDnsName": "ip-10-0-8-180.us-west-2.compute.internal",
    "PrivateIpAddress": "10.0.8.180",
    "PrivateIpAddresses": [
      {
        "Primary": true,
        "PrivateDnsName": "ip-10-0-8-180.us-west-2.compute.internal",
        "PrivateIpAddress": "10.0.8.180"
      }
    ]
  }
}

```

```

    ],
    "RequesterId": "AIDA4Z3Y7GSXTMEXAMPLE",
    "RequesterManaged": false,
    "SourceDestCheck": true,
    "Status": "pending",
    "SubnetId": "subnet-00a24d0d67acf6333",
    "TagSet": [],
    "VpcId": "vpc-02723a0feeeb9d57b"
  }
}

```

Para obtener más información, consulte [Interfaces de red elásticas](#) en la Guía del usuario de Amazon EC2.

- Para obtener información sobre la API, consulte [CreateNetworkInterface](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: En este ejemplo se crea la interfaz de red especificada.

```

New-EC2NetworkInterface -SubnetId subnet-1a2b3c4d -Description "my network
interface" -Group sg-12345678 -PrivateIpAddress 10.0.0.17

```

### Salida:

```

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
PrivateAddresses : {}
RequesterId      :
RequesterManaged : False
SourceDestCheck  : True

```

```
Status           : pending
SubnetId         : subnet-1a2b3c4d
TagSet           : {}
VpcId            : vpc-12345678
```

- Para obtener información sobre la API, consulte [CreateNetworkInterface](#) la referencia del AWS Tools for PowerShell cmdlet.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **CreatePlacementGroup** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `CreatePlacementGroup`.

### CLI

#### AWS CLI

Para crear un grupo de ubicaciones

Este comando de ejemplo crea un grupo de ubicación con el nombre especificado.

Comando:

```
aws ec2 create-placement-group --group-name my-cluster --strategy cluster
```

Para crear un grupo de colocación de particiones

Este comando de ejemplo crea un grupo de colocación de particiones denominado `HDFS-Group-A` con cinco particiones.

Comando:

```
aws ec2 create-placement-group --group-name HDFS-Group-A --strategy partition --partition-count 5
```

- Para obtener más información sobre la API, consulte [CreatePlacementGroup](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: En este ejemplo se crea un grupo de ubicación con el nombre especificado.

```
New-EC2PlacementGroup -GroupName my-placement-group -Strategy cluster
```

- Para obtener más información sobre la API, consulte [CreatePlacementGroup](#) la referencia del AWS Tools for PowerShell cmdlet.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **CreateRoute** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar CreateRoute.

### CLI

#### AWS CLI

Para crear una ruta

En este ejemplo, se crea una ruta para la tabla de rutas especificada. La ruta coincide con todo el tráfico IPv4 (0.0.0.0/0) y lo enruta a la puerta de enlace de Internet especificada. Si el comando se ejecuta correctamente, no se muestra ningún resultado.

Comando:

```
aws ec2 create-route --route-table-id rtb-22574640 --destination-cidr-block 0.0.0.0/0 --gateway-id igw-c0a643a9
```

Este comando de ejemplo crea una ruta en la tabla de rutas rtb-g8ff4ea2. La ruta coincide con el tráfico del bloque CIDR de IPv4 10.0.0.0/16 y lo enruta a la conexión de emparejamiento de VPC, pcx-111aaa22. Esta ruta permite que el tráfico se dirija a la VPC homóloga en la conexión de interconexión de VPC. Si el comando se ejecuta correctamente, no se muestra ningún resultado.

Comando:



```
aws ec2 create-route --route-table-id rtb-g8ff4ea2 --destination-cidr-block
10.0.0.0/16 --vpc-peering-connection-id pcx-1a2b3c4d
```

En este ejemplo, se crea una ruta en la tabla de rutas especificada que coincide con todo el tráfico de IPv6 (: :/0) y la enruta a la puerta de enlace de Internet de solo salida especificada.

Comando:

```
aws ec2 create-route --route-table-id rtb-dce620b8 --destination-ipv6-cidr-
block ::/0 --egress-only-internet-gateway-id eigw-01eadbd45ecd7943f
```

- Para obtener más información sobre la API, consulte [CreateRoute](#) la Referencia de comandos.AWS CLI

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: Este ejemplo crea la ruta especificada para la tabla de rutas especificada. La ruta coincide con todo el tráfico y lo envía a la puerta de enlace de Internet especificada.

```
New-EC2Route -RouteTableId rtb-1a2b3c4d -DestinationCidrBlock 0.0.0.0/0 -
GatewayId igw-1a2b3c4d
```

Salida:

```
True
```

- Para obtener más información sobre la API, consulte [CreateRoute](#) la referencia de AWS Tools for PowerShell cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **CreateRouteTable** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `CreateRouteTable`.

## CLI

### AWS CLI

#### Crear una tabla de enrutamiento

En este ejemplo, se crea una tabla de enrutamiento para la VPC especificada.

Comando:

```
aws ec2 create-route-table --vpc-id vpc-a01106c2
```

Salida:

```
{
  "RouteTable": {
    "Associations": [],
    "RouteTableId": "rtb-22574640",
    "VpcId": "vpc-a01106c2",
    "PropagatingVgws": [],
    "Tags": [],
    "Routes": [
      {
        "GatewayId": "local",
        "DestinationCidrBlock": "10.0.0.0/16",
        "State": "active"
      }
    ]
  }
}
```

- Para obtener más información sobre la API, consulte [CreateRouteTable](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: en este ejemplo se crea una tabla de enrutamiento para la VPC especificada.

```
New-EC2RouteTable -VpcId vpc-12345678
```

**Salida:**

```
Associations      : {}
PropagatingVgws  : {}
Routes           : {}
RouteTableId     : rtb-1a2b3c4d
Tags             : {}
VpcId            : vpc-12345678
```

- Para obtener información sobre la API, consulte [CreateRouteTable](#) la referencia del AWS Tools for PowerShell cmdlet.

**Ruby****SDK para Ruby****Note**

Hay más información al respecto. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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}'."
  return true
rescue StandardError => e
  puts "Error creating or associating route table: #{e.message}"
  puts "If the route table was created but not associated, you should " \
    "clean up by deleting the route table."

```

```
    return false
end

# Example usage:
def run_me
  vpc_id = ""
  subnet_id = ""
  gateway_id = ""
  destination_cidr_block = ""
  tag_key = ""
  tag_value = ""
  region = ""
  # Print usage information and then stop.
  if ARGV[0] == "--help" || ARGV[0] == "-h"
    puts "Usage: ruby ec2-ruby-example-create-route-table.rb " \
      "VPC_ID SUBNET_ID GATEWAY_ID DESTINATION_CIDR_BLOCK " \
      "TAG_KEY TAG_VALUE REGION"
  # Replace us-west-2 with the AWS Region you're using for Amazon EC2.
  puts "Example: ruby ec2-ruby-example-create-route-table.rb " \
    "vpc-0b6f769731EXAMPLE subnet-03d9303b57EXAMPLE igw-06ca90c011EXAMPLE " \
    "'0.0.0.0/0' my-key my-value us-west-2"
  exit 1
  # If no values are specified at the command prompt, use these default values.
  elsif ARGV.count.zero?
    vpc_id = "vpc-0b6f769731EXAMPLE"
    subnet_id = "subnet-03d9303b57EXAMPLE"
    gateway_id = "igw-06ca90c011EXAMPLE"
    destination_cidr_block = "0.0.0.0/0"
    tag_key = "my-key"
    tag_value = "my-value"
    # Replace us-west-2 with the AWS Region you're using for Amazon EC2.
    region = "us-west-2"
  # Otherwise, use the values as specified at the command prompt.
  else
    vpc_id = ARGV[0]
    subnet_id = ARGV[1]
    gateway_id = ARGV[2]
    destination_cidr_block = ARGV[3]
    tag_key = ARGV[4]
    tag_value = ARGV[5]
    region = ARGV[6]
  end

  ec2_resource = Aws::EC2::Resource.new(region: region)
```

```
if route_table_created_and_associated?(
  ec2_resource,
  vpc_id,
  subnet_id,
  gateway_id,
  destination_cidr_block,
  tag_key,
  tag_value
)
  puts "Route table created and associated."
else
  puts "Route table not created or not associated."
end
end

run_me if $PROGRAM_NAME == __FILE__
```

- Para obtener más información sobre la API, consulta [CreateRouteTable](#) la Referencia AWS SDK for Ruby de la API.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte [Cree recursos de Amazon EC2 mediante un SDK AWS](#). En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **CreateSecurityGroup** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `CreateSecurityGroup`.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en contexto en el siguiente ejemplo de código:

- [Comience a utilizar instancias](#)

## .NET

### AWS SDK for .NET

#### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

```
/// <summary>
/// Create an Amazon EC2 security group.
/// </summary>
/// <param name="groupName">The name for the new security group.</param>
/// <param name="groupDescription">A description of the new security group.</
param>
/// <returns>The group Id of the new security group.</returns>
public async Task<string> CreateSecurityGroup(string groupName, string
groupDescription)
{
    var response = await _amazonEC2.CreateSecurityGroupAsync(
        new CreateSecurityGroupRequest(groupName, groupDescription));

    return response.GroupId;
}
```

- Para obtener más información sobre la API, consulta [CreateSecurityGroup](#) la Referencia AWS SDK for .NET de la API.

## Bash

### AWS CLI con el script Bash

#### Note

Hay más información. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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
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
}

```

Las funciones de utilidad utilizadas en este ejemplo.

```

#####
# 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 obtener más información sobre la API, consulta [CreateSecurityGroup](#) la Referencia de AWS CLI comandos.

## C++

### SDK para C++

#### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

```
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;
```

- Para obtener más información sobre la API, consulta [CreateSecurityGroup](#) la Referencia AWS SDK for C++ de la API.

## CLI

### AWS CLI

Crear un grupo de seguridad para EC2-Classic

En este ejemplo, se crea un grupo de seguridad denominado MySecurityGroup.

**Comando:**

```
aws ec2 create-security-group --group-name MySecurityGroup --description "My security group"
```

**Salida:**

```
{
  "GroupId": "sg-903004f8"
}
```

**Crear un grupo de seguridad para EC2-VPC**

En este ejemplo, se crea un grupo de seguridad para la VPC especificada denominado MySecurityGroup.

**Comando:**

```
aws ec2 create-security-group --group-name MySecurityGroup --description "My security group" --vpc-id vpc-1a2b3c4d
```

**Salida:**

```
{
  "GroupId": "sg-903004f8"
}
```

Para obtener más información, consulte [Uso de los grupos de seguridad](#) en la Guía del usuario de la Interfaz de la línea de comandos de AWS .

- Para obtener más información sobre la API, consulta [CreateSecurityGroup](#) la Referencia de AWS CLI comandos.

## Java

## SDK para Java 2.x

 Note

Hay más información al respecto en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

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

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

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

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

```

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

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

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

```

- Para obtener más información sobre la API, consulta [CreateSecurityGroup](#) la Referencia AWS SDK for Java 2.x de la API.

## JavaScript

### SDK para JavaScript (v3)

#### Note

Hay más información. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

```

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

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

export const main = async () => {
    const command = new CreateSecurityGroupCommand({
        // Up to 255 characters in length. Cannot start with sg-.
        GroupName: "SECURITY_GROUP_NAME",
        // Up to 255 characters in length.
    });
}

```

```
    Description: "DESCRIPTION",
  });

  try {
    const { GroupId } = await client.send(command);
    console.log(GroupId);
  } catch (err) {
    console.error(err);
  }
};
```

- Para obtener más información sobre la API, consulta [CreateSecurityGroup](#) la Referencia AWS SDK for JavaScript de la API.

## Kotlin

### SDK para Kotlin

#### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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 obtener más información sobre la API, consulta [CreateSecurityGroup](#) la referencia sobre el AWS SDK para la API de Kotlin.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: en este ejemplo se crea un grupo de seguridad para la VPC especificada.



```
New-EC2SecurityGroup -GroupName my-security-group -Description "my security
group" -VpcId vpc-12345678
```

Salida:

```
sg-12345678
```

Ejemplo 2: En este ejemplo se crea un grupo de seguridad para EC2-Classic.

```
New-EC2SecurityGroup -GroupName my-security-group -Description "my security
group"
```

Salida:

```
sg-45678901
```

- Para obtener más información sobre la API, consulte la referencia de [CreateSecurityGroup AWS Tools for PowerShell](#) cmdlets.

## Python

### SDK para Python (Boto3)

#### Note

Hay más información al respecto. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

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

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

```

    :param security_group: A Boto3 SecurityGroup object. This is a high-level
object
                                that wraps security group actions.
    """
    self.ec2_resource = ec2_resource
    self.security_group = security_group

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

    def create(self, group_name, group_description):
        """
        Creates a security group in the default virtual private cloud (VPC) of
the
        current account.

        :param group_name: The name of the security group to create.
        :param group_description: The description of the security group to
create.
        :return: A Boto3 SecurityGroup object that represents the newly created
security group.
        """
        try:
            self.security_group = self.ec2_resource.create_security_group(
                GroupName=group_name, Description=group_description
            )
        except ClientError as err:
            logger.error(
                "Couldn't create security group %s. Here's why: %s: %s",
                group_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return self.security_group
```

- Para obtener más información sobre la API, consulta [CreateSecurityGroup](#) la AWS Referencia de API de SDK for Python (Boto3).

## Ruby

### SDK para Ruby

#### Note

Hay más información al respecto. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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}'."
  return security_group.group_id
rescue StandardError => e
  puts "Error creating security group: #{e.message}"
  return "Error"
end

# Adds an inbound rule to an Amazon Elastic Compute Cloud (Amazon EC2)
# security group.
#
# Prerequisites:
#
# - The security group.
#
# @param ec2_client [Aws::EC2::Client] An initialized Amazon EC2 client.
# @param security_group_id [String] The ID of the security group.
# @param ip_protocol [String] The network protocol for the inbound rule.
# @param from_port [String] The originating port for the inbound rule.
# @param to_port [String] The destination port for the inbound rule.
# @param cidr_ip_range [String] The CIDR IP range for the inbound rule.
# @return
# @example
#   exit 1 unless security_group_ingress_authorized?(
#     Aws::EC2::Client.new(region: 'us-west-2'),
#     'sg-030a858e078f1b9EX',
#     'tcp',
#     '80',
#     '80',
#     '0.0.0.0/0'
#   )
def security_group_ingress_authorized?(
  ec2_client,

```

```

    security_group_id,
    ip_protocol,
    from_port,
    to_port,
    cidr_ip_range
  )
  ec2_client.authorize_security_group_ingress(
    group_id: security_group_id,
    ip_permissions: [
      {
        ip_protocol: ip_protocol,
        from_port: from_port,
        to_port: to_port,
        ip_ranges: [
          {
            cidr_ip: cidr_ip_range
          }
        ]
      }
    ]
  )
  puts "Added inbound rule to security group '#{security_group_id}' for protocol
" \
    "'#{ip_protocol}' from port '#{from_port}' to port '#{to_port}' " \
    "with CIDR IP range '#{cidr_ip_range}'."
  return true
rescue StandardError => e
  puts "Error adding inbound rule to security group: #{e.message}"
  return false
end

# Displays information about a security group's IP permissions set in
# Amazon Elastic Compute Cloud (Amazon EC2).
#
# Prerequisites:
#
# - A security group with inbound rules, outbound rules, or both.
#
# @param p [Aws::EC2::Types::IpPermission] The IP permissions set.
# @example
#   ec2_client = Aws::EC2::Client.new(region: 'us-west-2')
#   response = ec2_client.describe_security_groups
#   unless sg.ip_permissions.empty?
#     describe_security_group_permissions(

```

```

#     response.security_groups[0].ip_permissions[0]
#   )
# end
def describe_security_group_permissions(perm)
  print " Protocol: #{perm.ip_protocol == '-1' ? 'All' : perm.ip_protocol}"

  unless perm.from_port.nil?
    if perm.from_port == "-1" || perm.from_port == -1
      print ", From: All"
    else
      print ", From: #{perm.from_port}"
    end
  end

  unless perm.to_port.nil?
    if perm.to_port == "-1" || perm.to_port == -1
      print ", To: All"
    else
      print ", To: #{perm.to_port}"
    end
  end

  if perm.key?(:ipv_6_ranges) && perm.ipv_6_ranges.count.positive?
    print ", CIDR IPv6: #{perm.ipv_6_ranges[0].cidr_ipv_6}"
  end

  if perm.key?(:ip_ranges) && perm.ip_ranges.count.positive?
    print ", CIDR IPv4: #{perm.ip_ranges[0].cidr_ip}"
  end

  print "\n"
end

# Displays information about available security groups in
# Amazon Elastic Compute Cloud (Amazon EC2).
#
# @param ec2_client [Aws::EC2::Client] An initialized Amazon EC2 client.
# @example
#   describe_security_groups(Aws::EC2::Client.new(region: 'us-west-2'))
def describe_security_groups(ec2_client)
  response = ec2_client.describe_security_groups

  if response.security_groups.count.positive?
    response.security_groups.each do |sg|

```

```
puts "-" * (sg.group_name.length + 13)
puts "Name:      #{sg.group_name}"
puts "Description: #{sg.description}"
puts "Group ID:   #{sg.group_id}"
puts "Owner ID:   #{sg.owner_id}"
puts "VPC ID:     #{sg.vpc_id}"

if sg.tags.count.positive?
  puts "Tags:"
  sg.tags.each do |tag|
    puts "  Key: #{tag.key}, Value: #{tag.value}"
  end
end

unless sg.ip_permissions.empty?
  puts "Inbound rules:" if sg.ip_permissions.count.positive?
  sg.ip_permissions.each do |p|
    describe_security_group_permissions(p)
  end
end

unless sg.ip_permissions_egress.empty?
  puts "Outbound rules:" if sg.ip_permissions.count.positive?
  sg.ip_permissions_egress.each do |p|
    describe_security_group_permissions(p)
  end
end
end
else
  puts "No security groups found."
end
rescue StandardError => e
  puts "Error getting information about security groups: #{e.message}"
end

# Deletes an Amazon Elastic Compute Cloud (Amazon EC2)
# security group.
#
# Prerequisites:
#
# - The security group.
#
# @param ec2_client [Aws::EC2::Client] An initialized
# Amazon EC2 client.
```

```

# @param security_group_id [String] The ID of the security group to delete.
# @return [Boolean] true if the security group was deleted; otherwise, false.
# @example
#   exit 1 unless security_group_deleted?(
#     Aws::EC2::Client.new(region: 'us-west-2'),
#     'sg-030a858e078f1b9EX'
#   )
def security_group_deleted?(ec2_client, security_group_id)
  ec2_client.delete_security_group(group_id: security_group_id)
  puts "Deleted security group '#{security_group_id}'."
  return true
rescue StandardError => e
  puts "Error deleting security group: #{e.message}"
  return false
end

# Example usage:
def run_me
  group_name = ""
  description = ""
  vpc_id = ""
  ip_protocol_http = ""
  from_port_http = ""
  to_port_http = ""
  cidr_ip_range_http = ""
  ip_protocol_ssh = ""
  from_port_ssh = ""
  to_port_ssh = ""
  cidr_ip_range_ssh = ""
  region = ""
  # Print usage information and then stop.
  if ARGV[0] == "--help" || ARGV[0] == "-h"
    puts "Usage:  ruby ec2-ruby-example-security-group.rb " \
      "GROUP_NAME DESCRIPTION VPC_ID IP_PROTOCOL_1 FROM_PORT_1 TO_PORT_1 " \
      "CIDR_IP_RANGE_1 IP_PROTOCOL_2 FROM_PORT_2 TO_PORT_2 " \
      "CIDR_IP_RANGE_2 REGION"
    puts "Example: ruby ec2-ruby-example-security-group.rb " \
      "my-security-group 'This is my security group.' vpc-6713dfEX " \
      "tcp 80 80 '0.0.0.0/0' tcp 22 22 '0.0.0.0/0' us-west-2"
    exit 1
  # If no values are specified at the command prompt, use these default values.
  elsif ARGV.count.zero?
    group_name = "my-security-group"
    description = "This is my security group."

```



```
vpc_id = "vpc-6713dfEX"
ip_protocol_http = "tcp"
from_port_http = "80"
to_port_http = "80"
cidr_ip_range_http = "0.0.0.0/0"
ip_protocol_ssh = "tcp"
from_port_ssh = "22"
to_port_ssh = "22"
cidr_ip_range_ssh = "0.0.0.0/0"
# Replace us-west-2 with the AWS Region you're using for Amazon EC2.
region = "us-west-2"
# Otherwise, use the values as specified at the command prompt.
else
  group_name = ARGV[0]
  description = ARGV[1]
  vpc_id = ARGV[2]
  ip_protocol_http = ARGV[3]
  from_port_http = ARGV[4]
  to_port_http = ARGV[5]
  cidr_ip_range_http = ARGV[6]
  ip_protocol_ssh = ARGV[7]
  from_port_ssh = ARGV[8]
  to_port_ssh = ARGV[9]
  cidr_ip_range_ssh = ARGV[10]
  region = ARGV[11]
end

security_group_id = ""
security_group_exists = false
ec2_client = Aws::EC2::Client.new(region: region)

puts "Attempting to create security group..."
security_group_id = create_security_group(
  ec2_client,
  group_name,
  description,
  vpc_id
)
if security_group_id == "Error"
  puts "Could not create security group. Skipping this step."
else
  security_group_exists = true
end
```

```
if security_group_exists
  puts "Attempting to add inbound rules to security group..."
  unless security_group_ingress_authorized?(
    ec2_client,
    security_group_id,
    ip_protocol_http,
    from_port_http,
    to_port_http,
    cidr_ip_range_http
  )
    puts "Could not add inbound HTTP rule to security group. " \
      "Skipping this step."
  end

  unless security_group_ingress_authorized?(
    ec2_client,
    security_group_id,
    ip_protocol_ssh,
    from_port_ssh,
    to_port_ssh,
    cidr_ip_range_ssh
  )
    puts "Could not add inbound SSH rule to security group. " \
      "Skipping this step."
  end
end

puts "\nInformation about available security groups:"
describe_security_groups(ec2_client)

if security_group_exists
  puts "\nAttempting to delete security group..."
  unless security_group_deleted?(ec2_client, security_group_id)
    puts "Could not delete security group. You must delete it yourself."
  end
end
end

run_me if $PROGRAM_NAME == __FILE__
```

- Para obtener más información sobre la API, consulta [CreateSecurityGroup](#) la Referencia AWS SDK for Ruby de la API.

## SAP ABAP

### SDK de SAP ABAP

#### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

```
TRY.
    oo_result = lo_ec2->createsecuritygroup(
        iv_description = 'Security group example'
        iv_groupname = iv_security_group_name
        iv_vpcid = iv_vpc_id
    ).
    MESSAGE 'Security group created.' TYPE 'I'.
CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
    DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
    MESSAGE lv_error TYPE 'E'.
ENDTRY.
```

- Para obtener más información sobre la API, consulte [CreateSecurityGroup](#) la referencia sobre la API ABAP del AWS SDK para SAP.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte [Cree recursos de Amazon EC2 mediante un SDK AWS](#). En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **CreateSnapshot** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar CreateSnapshot.

### CLI

#### AWS CLI

Para crear una instantánea

Este comando de ejemplo crea una instantánea del volumen con un identificador de volumen `vol-1234567890abcdef0` y una breve descripción para identificar la instantánea.

Comando:

```
aws ec2 create-snapshot --volume-id vol-1234567890abcdef0 --description "This is my root volume snapshot"
```

Salida:

```
{
  "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 crear una instantánea con etiquetas

Este comando de ejemplo crea una instantánea y aplica dos etiquetas: `purpose=prod` y `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}]'
```

Salida:

```
{
  "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 obtener más información sobre la API, consulte la Referencia de comandos. [CreateSnapshot](#) AWS CLI

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: En este ejemplo se crea una instantánea del volumen especificado.

```
New-EC2Snapshot -VolumeId vol-12345678 -Description "This is a test"
```

Salida:

```
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 obtener más información sobre la API, consulte [CreateSnapshot](#) la referencia del AWS Tools for PowerShell cmdlet.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **CreateSpotDatafeedSubscription** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `CreateSpotDatafeedSubscription`.

### CLI

#### AWS CLI

Para crear una fuente de datos de Spot Instance

En el siguiente `create-spot-datafeed-subscription` ejemplo, se crea una fuente de datos de instancias puntuales.

```
aws ec2 create-spot-datafeed-subscription \  
  --bucket my-bucket \  
  --prefix spot-data-feed
```

Salida:

```
{  
  "SpotDatafeedSubscription": {  
    "Bucket": "my-bucket",  
    "OwnerId": "123456789012",  
    "Prefix": "spot-data-feed",  
    "State": "Active"  
  }  
}
```

La fuente de datos se almacena en el bucket de Amazon S3 que especificó. Los nombres de los archivos de esta fuente de datos tienen el siguiente formato.

```
my-bucket.s3.amazonaws.com/spot-data-feed/123456789012.YYYY-MM-DD-  
HH.n.abcd1234.gz
```

Para obtener más información, consulte la [fuente de datos de instancias puntuales](#) en la Guía del usuario de Amazon Elastic Compute Cloud para instancias de Linux.

- Para obtener más información sobre la API, consulte [CreateSpotDatafeedSubscription](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: En este ejemplo se crea una fuente de datos de instancias puntuales.

```
New-EC2SpotDatafeedSubscription -Bucket my-s3-bucket -Prefix spotdata
```

Salida:

```
Bucket   : my-s3-bucket  
Fault    :  
OwnerId  : 123456789012  
Prefix   : spotdata  
State    : Active
```

- Para obtener más información sobre la API, consulte [CreateSpotDatafeedSubscription](#) la referencia de AWS Tools for PowerShell cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **CreateSubnet** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar CreateSubnet.

## CLI

### AWS CLI

#### Ejemplo 1: Crear una subred solo con un bloque de CIDR IPv4

En el siguiente ejemplo de `create-subnet`, se crea una subred en la VPC especificada con el bloque de CIDR IPv4 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}]
```

#### Salida:

```
{  
  "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"  
  }  
}
```

#### Ejemplo 2: Crear una subred con bloques de CIDR IPv4 e IPv6



En el siguiente ejemplo de `create-subnet`, se crea una subred en la VPC especificada con los bloques de CIDR IPv4 e IPv6 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}]
```

Salida:

```
{
  "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"
  }
}
```

```
}
```

### Ejemplo 3: Crear una subred solo con un bloque de CIDR IPv6

En el siguiente ejemplo de `create-subnet`, se crea una subred en la VPC especificada con el bloque de CIDR IPv6 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}]
```

Salida:

```
{  
  "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 obtener más información, consulte [VPC y subredes](#) en la Guía del usuario de Amazon VPC.

- Para obtener más información sobre la API, consulte [CreateSubnet](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: En este ejemplo se crea una subred con el CIDR especificado.

```
New-EC2Subnet -VpcId vpc-12345678 -CidrBlock 10.0.0.0/24
```

Salida:

```
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 obtener más información sobre la API, consulte la referencia del [CreateSubnet AWS Tools for PowerShell](#) cmdlet.

## Ruby

### SDK para Ruby

#### Note

Hay más información al respecto. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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_value [String] The value portion of the tag for the subnet.
# @return [Boolean] true if the subnet was created and tagged;
#   otherwise, false.
# @example
#   exit 1 unless subnet_created_and_tagged?(
#     Aws::EC2::Resource.new(region: 'us-west-2'),
#     'vpc-6713dfEX',
#     '10.0.0.0/24',
#     'us-west-2a',
#     'my-key',
#     'my-value'
#   )
def subnet_created_and_tagged?(
  ec2_resource,
```

```

    vpc_id,
    cidr_block,
    availability_zone,
    tag_key,
    tag_value
  )
  subnet = ec2_resource.create_subnet(
    vpc_id: vpc_id,
    cidr_block: cidr_block,
    availability_zone: availability_zone
  )
  subnet.create_tags(
    tags: [
      {
        key: tag_key,
        value: tag_value
      }
    ]
  )
  puts "Subnet created with ID '#{subnet.id}' in VPC with ID '#{vpc_id}' " \
    "and CIDR block '#{cidr_block}' in availability zone " \
    "'#{availability_zone}' and tagged with key '#{tag_key}' and " \
    "value '#{tag_value}'."
  return true
rescue StandardError => e
  puts "Error creating or tagging subnet: #{e.message}"
  return false
end

# Example usage:
def run_me
  vpc_id = ""
  cidr_block = ""
  availability_zone = ""
  tag_key = ""
  tag_value = ""
  region = ""
  # Print usage information and then stop.
  if ARGV[0] == "--help" || ARGV[0] == "-h"
    puts "Usage:  ruby ec2-ruby-example-create-subnet.rb " \
      "VPC_ID CIDR_BLOCK AVAILABILITY_ZONE TAG_KEY TAG_VALUE REGION"
    # Replace us-west-2 with the AWS Region you're using for Amazon EC2.
    puts "Example: ruby ec2-ruby-example-create-subnet.rb " \
      "vpc-6713dfEX 10.0.0.0/24 us-west-2a my-key my-value us-west-2"
  end
end

```

```
    exit 1
  # If no values are specified at the command prompt, use these default values.
  elsif ARGV.count.zero?
    vpc_id = "vpc-6713dfEX"
    cidr_block = "10.0.0.0/24"
    availability_zone = "us-west-2a"
    tag_key = "my-key"
    tag_value = "my-value"
    # Replace us-west-2 with the AWS Region you're using for Amazon EC2.
    region = "us-west-2"
  # Otherwise, use the values as specified at the command prompt.
  else
    vpc_id = ARGV[0]
    cidr_block = ARGV[1]
    availability_zone = ARGV[2]
    tag_key = ARGV[3]
    tag_value = ARGV[4]
    region = ARGV[5]
  end

  ec2_resource = Aws::EC2::Resource.new(region: region)

  if subnet_created_and_tagged?(
    ec2_resource,
    vpc_id,
    cidr_block,
    availability_zone,
    tag_key,
    tag_value
  )
    puts "Subnet created and tagged."
  else
    puts "Subnet not created or not tagged."
  end
end

run_me if $PROGRAM_NAME == __FILE__
```

- Para obtener más información sobre la API, consulta [CreateSubnet](#) la Referencia AWS SDK for Ruby de la API.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte [Cree recursos de Amazon EC2 mediante un SDK AWS](#). En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **CreateTags** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `CreateTags`.

C++

SDK para C++

### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

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

Aws::EC2::Model::Tag nameTag;
nameTag.SetKey("Name");
nameTag.SetValue(instanceName);

Aws::EC2::Model::CreateTagsRequest createRequest;
createRequest.AddResources(instanceID);
createRequest.AddTags(nameTag);

Aws::EC2::Model::CreateTagsOutcome createOutcome = ec2Client.CreateTags(
    createRequest);
if (!createOutcome.IsSuccess()) {
    std::cerr << "Failed to tag ec2 instance " << instanceID <<
        " with name " << instanceName << ":" <<
        createOutcome.GetError().GetMessage() << std::endl;
    return false;
}
```

- Para obtener más información sobre la API, consulta [CreateTags](#) la Referencia AWS SDK for C++ de la API.

## CLI

### AWS CLI

#### Ejemplo 1: Para añadir una etiqueta a un recurso

En el siguiente ejemplo de `create-tags`, se agrega la etiqueta `Stack=production` a la imagen especificada o se sobrescribe una etiqueta existente para la AMI en la que la clave de etiqueta es `Stack`.

```
aws ec2 create-tags \  
  --resources ami-1234567890abcdef0 \  
  --tags Key=Stack,Value=production
```

Para obtener más información, consulte [Este es el título del tema](#) en la Guía del usuario de Amazon Elastic Compute Cloud para instancias de Linux.

#### Ejemplo 2: Para añadir etiquetas a varios recursos

En el siguiente ejemplo de `create-tags`, se agregan (o sobrescriben) dos etiquetas para una AMI y una instancia. Una de las etiquetas tiene una clave (`webserver`), pero no tiene valor (el valor se establece en una cadena vacía). La otra etiqueta tiene una clave (`stack`) y un valor (`Production`).

```
aws ec2 create-tags \  
  --resources ami-1a2b3c4d i-1234567890abcdef0 \  
  --tags Key=webserver,Value=   Key=stack,Value=Production
```

Para obtener más información, consulte [Este es el título del tema](#) en la Guía del usuario de Amazon Elastic Compute Cloud para instancias de Linux.

#### Ejemplo 3: Para añadir etiquetas que contengan caracteres especiales

En el siguiente ejemplo de `create-tags`, se agrega la etiqueta `[Group]=test` a una instancia. Los corchetes (`[` y `]`) son caracteres especiales y deben incluirse en el carácter de escape. En los siguientes ejemplos también se usa el carácter de continuación de línea adecuado para cada entorno.

Si usa Windows, encierre el elemento que tiene caracteres especiales entre comillas dobles (`"`) y, a continuación, preceda cada carácter de comillas dobles con una barra invertida (`\`) de la siguiente manera:



```
aws ec2 create-tags ^
  --resources i-1234567890abcdef0 ^
  --tags Key=\"[Group]\",Value=test
```

Si utiliza Windows PowerShell, escriba el elemento con caracteres especiales entre comillas dobles («»), coloque una barra invertida (\) delante de cada carácter entre comillas dobles y, a continuación, rodee toda la estructura de claves y valores entre comillas simples ('), de la siguiente manera:

```
aws ec2 create-tags `
  --resources i-1234567890abcdef0 `
  --tags 'Key=\"[Group]\",Value=test'
```

Si usa Linux u OS X, encierre el elemento con caracteres especiales entre comillas dobles (") y, a continuación, encierre toda la estructura de clave y valor entre comillas simples (') de la siguiente manera:

```
aws ec2 create-tags \
  --resources i-1234567890abcdef0 \
  --tags 'Key="[Group]",Value=test'
```

Para obtener más información, consulte [Este es el título del tema](#) en la Guía del usuario de Amazon Elastic Compute Cloud para instancias de Linux.

- Para obtener más información sobre la API, consulte [CreateTags](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: en este ejemplo se agrega una sola etiqueta al recurso especificado. La clave de la etiqueta es 'myTag' y el valor de la etiqueta es 'myTagValue'. La sintaxis utilizada en este ejemplo requiere la PowerShell versión 3 o superior.

```
New-EC2Tag -Resource i-12345678 -Tag @{ Key="myTag"; Value="myTagValue" }
```

Ejemplo 2: Este ejemplo actualiza o agrega las etiquetas especificadas al recurso especificado. La sintaxis utilizada en este ejemplo requiere PowerShell la versión 3 o superior.

```
New-EC2Tag -Resource i-12345678 -Tag @( @{ Key="myTag"; Value="newTagValue" },
@{ Key="test"; Value="anotherTagValue" } )
```

Ejemplo 3: Con la PowerShell versión 2, debe usar `New-Object` para crear la etiqueta para el parámetro `Tag`.

```
$tag = New-Object Amazon.EC2.Model.Tag
$tag.Key = "myTag"
$tag.Value = "myTagValue"

New-EC2Tag -Resource i-12345678 -Tag $tag
```

- Para obtener más información sobre la API, consulte la referencia del [CreateTags AWS Tools for PowerShell](#) cmdlet.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **CreateVolume** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `CreateVolume`.

### CLI

#### AWS CLI

Para crear un volumen SSD de uso general (gp2) vacío

En el siguiente `create-volume` ejemplo, se crea un volumen SSD de uso general (gp2) de 80 GiB en la zona de disponibilidad especificada. Tenga en cuenta que la región actual debe ser `us-east-1`, o bien puede añadir el `--region` parámetro para especificarla para el comando.

```
aws ec2 create-volume \
  --volume-type gp2 \
  --size 80 \
  --availability-zone us-east-1a
```

Salida:

```
{
  "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
}
```

Si no especifica un tipo de volumen, el tipo de volumen predeterminado esgp2.

```
aws ec2 create-volume \
  --size 80 \
  --availability-zone us-east-1a
```

Ejemplo 2: Para crear un volumen SSD (io1) de IOPS aprovisionado a partir de una instantánea

En el siguiente create-volume ejemplo, se crea un volumen SSD de IOPS aprovisionadas (io1) con 1000 IOPS aprovisionadas en la zona de disponibilidad especificada mediante la instantánea especificada.

```
aws ec2 create-volume \
  --volume-type io1 \
  --iops 1000 \
  --snapshot-id snap-066877671789bd71b \
  --availability-zone us-east-1a
```

Salida:

```
{
  "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
}
```

### Ejemplo 3: Para crear un volumen cifrado

En el siguiente `create-volume` ejemplo, se crea un volumen cifrado con la CMK predeterminada para el cifrado de EBS. Si el cifrado está deshabilitado de forma predeterminada, debe especificar el `--encrypted` parámetro de la siguiente manera.

```
aws ec2 create-volume \
  --size 80 \
  --encrypted \
  --availability-zone us-east-1a
```

### Salida:

```
{
  "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
}
```

Si el cifrado está activado de forma predeterminada, el siguiente comando de ejemplo crea un volumen cifrado, incluso sin el `--encrypted` parámetro.

```
aws ec2 create-volume \
  --size 80 \
  --availability-zone us-east-1a
```

Si utiliza el `--kms-key-id` parámetro para especificar una CMK gestionada por el cliente, debe especificar el `--encrypted` parámetro incluso si el cifrado está activado de forma predeterminada.

```
aws ec2 create-volume \  
  --volume-type gp2 \  
  --size 80 \  
  --encrypted \  
  --kms-key-id 0ea3fef3-80a7-4778-9d8c-1c0c6EXAMPLE \  
  --availability-zone us-east-1a
```

#### Ejemplo 4: Para crear un volumen con etiquetas

En el siguiente `create-volume` ejemplo, se crea un volumen y se añaden dos etiquetas.

```
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 obtener más información sobre la API, consulte [CreateVolume](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: En este ejemplo se crea el volumen especificado.

```
New-EC2Volume -Size 50 -AvailabilityZone us-west-2a -VolumeType gp2
```

Salida:

```
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

```

Ejemplo 2: Esta solicitud de ejemplo crea un volumen y aplica una etiqueta con una clave de pila y un valor de producción.

```

$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 obtener más información sobre la API, consulte [CreateVolume](#) la referencia del AWS Tools for PowerShell cmdlet.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **CreateVpc** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar CreateVpc.

### CLI

#### AWS CLI

##### Ejemplo 1: Crear una VPC

En el siguiente ejemplo de `create-vpc`, se crea una VPC con el bloque de CIDR IPv4 especificado y una etiqueta de nombre.

```

aws ec2 create-vpc \
  --cidr-block 10.0.0.0/16 \

```

```
--tag-specifications ResourceType=vpc,Tags=[{Key=Name,Value=MyVpc}]
```

Salida:

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

Ejemplo 2: Crear una VPC con un tenencia dedicada

En el siguiente ejemplo de `create-vpc`, se crea una VPC con el bloque de CIDR IPv4 especificado y una tenencia dedicada.

```
aws ec2 create-vpc \
  --cidr-block 10.0.0.0/16 \
  --instance-tenancy dedicated
```

Salida:

```
{
  "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
  }
}
```

### Ejemplo 3: Crear una VPC con un bloque de CIDR IPv6

En el siguiente ejemplo de `create-vpc`, se crea una VPC con un bloque de CIDR IPv6 proporcionado por Amazon.

```
aws ec2 create-vpc \
  --cidr-block 10.0.0.0/16 \
  --amazon-provided-ipv6-cidr-block
```

Salida:

```
{
  "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
  }
}

```

#### Ejemplo 4: Crear una VPC con un CIDR desde un grupo de IPAM

En el siguiente ejemplo de `create-vpc`, se crea una VPC con un CIDR desde un grupo del Administrador de direcciones IP (IPAM) de Amazon VPC.

Linux y 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"}]
```

Salida:

```
{
  "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 obtener más información, consulte [Crear una VPC que use un CIDR de un grupo de IPAM](#) en la Guía del usuario de IPAM de Amazon VPC.

- Para obtener más información sobre la API, consulte [CreateVpcla Referencia de AWS CLI comandos](#).

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: en este ejemplo se crea una VPC con el CIDR especificado. Amazon VPC también crea lo siguiente para la VPC: un conjunto de opciones de DHCP predeterminado, una tabla de enrutamiento principal y una ACL de red predeterminada.

```
New-EC2VPC -CidrBlock 10.0.0.0/16
```

### Salida:

```
CidrBlock      : 10.0.0.0/16
DhcpOptionsId  : dopt-1a2b3c4d
InstanceTenancy : default
IsDefault      : False
State          : pending
Tags           : {}
VpcId          : vpc-12345678
```

- Para obtener más información sobre la API, consulte la referencia de [CreateVpccmdlets](#) AWS Tools for PowerShell .

## Ruby

### SDK para Ruby

#### Note

Hay más información al respecto. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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}'."
  return true
rescue StandardError => e
  puts "#{e.message}"
  return false
end

# Example usage:
def run_me
  cidr_block = ""
  tag_key = ""
  tag_value = ""
  region = ""
  # Print usage information and then stop.
  if ARGV[0] == "--help" || ARGV[0] == "-h"

```

```
puts "Usage:  ruby ec2-ruby-example-create-vpc.rb " \
      "CIDR_BLOCK TAG_KEY TAG_VALUE REGION"
# Replace us-west-2 with the AWS Region you're using for Amazon EC2.
puts "Example: ruby ec2-ruby-example-create-vpc.rb " \
      "10.0.0.0/24 my-key my-value us-west-2"
exit 1
# If no values are specified at the command prompt, use these default values.
elsif ARGV.count.zero?
  cidr_block = "10.0.0.0/24"
  tag_key = "my-key"
  tag_value = "my-value"
  # Replace us-west-2 with the AWS Region you're using for Amazon EC2.
  region = "us-west-2"
# Otherwise, use the values as specified at the command prompt.
else
  cidr_block = ARGV[0]
  tag_key = ARGV[1]
  tag_value = ARGV[2]
  region = ARGV[3]
end

ec2_resource = Aws::EC2::Resource.new(region: region)

if vpc_created_and_tagged?(
  ec2_resource,
  cidr_block,
  tag_key,
  tag_value
)
  puts "VPC created and tagged."
else
  puts "VPC not created or not tagged."
end
end

run_me if $PROGRAM_NAME == __FILE__
```

- Para obtener más información sobre la API, consulta [CreateVpca](#) Referencia AWS SDK for Ruby de la API.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte [Cree recursos de Amazon EC2 mediante un SDK AWS](#). En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **CreateVpcEndpoint** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `CreateVpcEndpoint`.

### CLI

#### AWS CLI

Ejemplo 1: Para crear un punto final de puerta de enlace

En el siguiente `create-vpc-endpoint` ejemplo, se crea un punto de enlace de VPC entre VPC y Amazon `vpc-1a2b3c4d` S3 en la `us-east-1` región y se asocia la tabla `rtb-11aa22bb` de enrutamiento al punto de enlace.

```
aws ec2 create-vpc-endpoint \
  --vpc-id vpc-1a2b3c4d \
  --service-name com.amazonaws.us-east-1.s3 \
  --route-table-ids rtb-11aa22bb
```

Salida:

```
{
  "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 obtener más información, consulte [Crear un punto de enlace de puerta de enlace](#) de enlace en la AWS PrivateLink guía.

Ejemplo 2: Para crear un punto final de interfaz

El siguiente `create-vpc-endpoint` ejemplo crea un punto de enlace de VPC de interfaz entre VPC y Amazon `vpc-1a2b3c4d` S3 en la región. `us-east-1` El comando crea el punto final en la subred `subnet-1a2b3c4d`, lo asocia al grupo `sg-1a2b3c4d` de seguridad y añade una etiqueta con la clave «Servicio» y el valor «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}]
```

Salida:

```
{
  "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 obtener más información, consulte [Creación de un punto final de interfaz](#) en la Guía del usuario de AWS PrivateLink.

**Ejemplo 3:** Para crear un punto final de Gateway Load Balancer

En el siguiente `create-vpc-endpoint` ejemplo, se crea un punto final de Gateway Load Balancer entre la VPC `vpc-111122223333aabbc` y un servicio que se configura mediante un 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-111122223333aabbc \
  --subnet-ids subnet-0011aabbcc2233445

```

Salida:

```

{
  "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 obtener más información, consulte los [puntos finales del Gateway Load Balancer](#) en la Guía del usuario de AWS PrivateLink

- Para obtener más información sobre la API, consulte la Referencia [CreateVpcEndpoint](#) de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: En este ejemplo, se crea un nuevo punto de enlace de VPC para el servicio `com.amazonaws.eu-west-1.s3` en la VPC `vpc-0fc1ff23f45b678eb`

```
New-EC2VpcEndpoint -ServiceName com.amazonaws.eu-west-1.s3 -VpcId
vpc-0fc1ff23f45b678eb
```

Salida:

```
ClientToken VpcEndpoint
-----
                Amazon.EC2.Model.VpcEndpoint
```

- Para obtener más [CreateVpcEndpoint AWS Tools for PowerShell](#) información sobre la API, consulte la referencia del cmdlet.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo `CreateVpnConnection` con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `CreateVpnConnection`.

### CLI

#### AWS CLI

Ejemplo 1: Para crear una conexión VPN con enrutamiento dinámico

El siguiente `create-vpn-connection` ejemplo crea una conexión VPN entre la puerta de enlace privada virtual especificada y la puerta de enlace del cliente especificada, y aplica etiquetas a la conexión VPN. El resultado incluye la información de configuración del dispositivo de puerta de enlace del cliente, en formato XML.

```
aws ec2 create-vpn-connection \
  --type ipsec.1 \
  --customer-gateway-id cgw-001122334455aabbc \
  --vpn-gateway-id vgw-1a1a1a1a1a1a2b2b2 \
  --tag-specification 'ResourceType=vpn-connection,Tags=[{Key=Name,Value=BGP-VPN}]'
```

Salida:

```
{
  "VpnConnection": {
    "CustomerGatewayConfiguration": "...configuration information...",
    "CustomerGatewayId": "cgw-001122334455aabbc",
    "Category": "VPN",
    "State": "pending",
    "VpnConnectionId": "vpn-123123123123abcab",
    "VpnGatewayId": "vgw-1a1a1a1a1a1a2b2b2",
    "Options": {
      "EnableAcceleration": false,
      "StaticRoutesOnly": false,
      "LocalIpv4NetworkCidr": "0.0.0.0/0",
      "RemoteIpv4NetworkCidr": "0.0.0.0/0",
      "TunnelInsideIpVersion": "ipv4",
```

```

        "TunnelOptions": [
            {},
            {}
        ]
    },
    "Routes": [],
    "Tags": [
        {
            "Key": "Name",
            "Value": "BGP-VPN"
        }
    ]
}
}

```

Para obtener más información, consulte [Cómo funciona la VPN de AWS sitio a sitio en la Guía del usuario de VPN](#) de sitio a sitio.AWS

Ejemplo 2: Para crear una conexión VPN con enrutamiento estático

El siguiente `create-vpn-connection` ejemplo crea una conexión VPN entre la puerta de enlace privada virtual especificada y la puerta de enlace del cliente especificada. Las opciones especifican el enrutamiento estático. El resultado incluye la información de configuración del dispositivo de pasarela de clientes, en formato XML.

```

aws ec2 create-vpn-connection \
  --type ipsec.1 \
  --customer-gateway-id cgw-001122334455aabbc \
  --vpn-gateway-id vgw-1a1a1a1a1a1a2b2b2 \
  --options "{\"StaticRoutesOnly\":true}"

```

Salida:

```

{
  "VpnConnection": {
    "CustomerGatewayConfiguration": "..configuration information...",
    "CustomerGatewayId": "cgw-001122334455aabbc",
    "Category": "VPN",
    "State": "pending",
    "VpnConnectionId": "vpn-123123123123abcab",
    "VpnGatewayId": "vgw-1a1a1a1a1a1a2b2b2",
    "Options": {

```

```

    "EnableAcceleration": false,
    "StaticRoutesOnly": true,
    "LocalIpv4NetworkCidr": "0.0.0.0/0",
    "RemoteIpv4NetworkCidr": "0.0.0.0/0",
    "TunnelInsideIpVersion": "ipv4",
    "TunnelOptions": [
        {},
        {}
    ]
  },
  "Routes": [],
  "Tags": []
}
}

```

Para obtener más información, consulte [Cómo funciona la VPN de AWS sitio a sitio en la Guía del usuario de VPN](#) de sitio a sitio.AWS

Ejemplo 3: Para crear una conexión VPN y especificar la suya propia dentro del CIDR y de una clave previamente compartida

En el siguiente `create-vpn-connection` ejemplo, se crea una conexión VPN y se especifica la dirección IP interna, el bloque CIDR, y una clave personalizada previamente compartida para cada túnel. Los valores especificados se devuelven en la `CustomerGatewayConfiguration` información.

```

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}]'

```

Salida:

```

{
  "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 obtener más información, consulte [Cómo funciona la VPN de AWS sitio a sitio en la Guía del usuario de VPN](#) de sitio a sitio.AWS

Ejemplo 4: Para crear una conexión VPN que admita el tráfico IPv6

El siguiente `create-vpn-connection` ejemplo crea una conexión VPN que admite el tráfico IPv6 entre la puerta de enlace de tránsito especificada y la puerta de enlace del cliente especificada. Las opciones de túnel para ambos túneles especifican que AWS debe iniciar la negociación del 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}]

```

**Salida:**

```
{
  "VpnConnection": {
    "CustomerGatewayConfiguration": "..configuration information...",
    "CustomerGatewayId": "cgw-001122334455aabbcc",
    "Category": "VPN",
    "State": "pending",
    "VpnConnectionId": "vpn-111111111122222222",
    "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 obtener más información, consulte [Cómo funciona la VPN de AWS sitio a sitio en la Guía del usuario de VPN](#) de sitio a sitio.AWS

- Para obtener más información sobre la API, consulte [CreateVpnConnection](#) la Referencia de comandos.AWS CLI

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: Este ejemplo crea una conexión VPN entre la puerta de enlace privada virtual especificada y la puerta de enlace del cliente especificada. El resultado incluye la información de configuración que necesita el administrador de red, en formato XML.

```
New-EC2VpnConnection -Type ipsec.1 -CustomerGatewayId cgw-1a2b3c4d -VpnGatewayId vgw-1a2b3c4d
```

#### Salida:

```
CustomerGatewayConfiguration : [XML document]
CustomerGatewayId            : cgw-1a2b3c4d
Options                       :
Routes                       : {}
State                        : pending
Tags                         : {}
Type                         :
VgwTelemetry                 : {}
VpnConnectionId             : vpn-12345678
VpnGatewayId                 : vgw-1a2b3c4d
```

Ejemplo 2: en este ejemplo se crea la conexión VPN y se captura la configuración en un archivo con el nombre especificado.

```
(New-EC2VpnConnection -CustomerGatewayId cgw-1a2b3c4d -VpnGatewayId vgw-1a2b3c4d).CustomerGatewayConfiguration | Out-File C:\path\vpn-configuration.xml
```

Ejemplo 3: Este ejemplo crea una conexión VPN, con enrutamiento estático, entre la puerta de enlace privada virtual especificada y la puerta de enlace del cliente especificada.

```
New-EC2VpnConnection -Type ipsec.1 -CustomerGatewayId cgw-1a2b3c4d -VpnGatewayId vgw-1a2b3c4d -Options_StaticRoutesOnly $true
```

- Para obtener más información sobre la API, consulte [CreateVpnConnection](#) la referencia de AWS Tools for PowerShell cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **CreateVpnConnectionRoute** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `CreateVpnConnectionRoute`.

### CLI

#### AWS CLI

Para crear una ruta estática para una conexión VPN

En este ejemplo, se crea una ruta estática para la conexión VPN especificada. Si el comando se ejecuta correctamente, no se muestra ningún resultado.

Comando:

```
aws ec2 create-vpn-connection-route --vpn-connection-id vpn-40f41529 --
destination-cidr-block 11.12.0.0/16
```

- Para obtener más información sobre la API, consulte [CreateVpnConnectionRoute](#) la Referencia de AWS CLI comandos.

### PowerShell

#### Herramientas para PowerShell

Ejemplo 1: Este ejemplo crea la ruta estática especificada para la conexión VPN especificada.

```
New-EC2VpnConnectionRoute -VpnConnectionId vpn-12345678 -DestinationCidrBlock
11.12.0.0/16
```

- Para obtener más información sobre la API, consulte [CreateVpnConnectionRoute](#) la referencia de AWS Tools for PowerShell cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.



## Úselo `CreateVpnGateway` con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `CreateVpnGateway`.

### CLI

#### AWS CLI

Para crear una puerta de enlace privada virtual

En este ejemplo se crea una puerta de enlace privada virtual.

Comando:

```
aws ec2 create-vpn-gateway --type ipsec.1
```

Salida:

```
{
  "VpnGateway": {
    "AmazonSideAsn": 64512,
    "State": "available",
    "Type": "ipsec.1",
    "VpnGatewayId": "vgw-9a4cacf3",
    "VpcAttachments": []
  }
}
```

Para crear una puerta de enlace privada virtual con un ASN específico del lado de Amazon

En este ejemplo, se crea una puerta de enlace privada virtual y se especifica el número de sistema autónomo (ASN) para la parte Amazon de la sesión de BGP.

Comando:

```
aws ec2 create-vpn-gateway --type ipsec.1 --amazon-side-asn 65001
```

Salida:

```
{
```

```
"VpnGateway": {
  "AmazonSideAsn": 65001,
  "State": "available",
  "Type": "ipsec.1",
  "VpnGatewayId": "vgw-9a4cacf3",
  "VpcAttachments": []
}
```

- Para obtener más información sobre la API, consulte la Referencia [CreateVpnGateway](#) de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: En este ejemplo se crea la puerta de enlace privada virtual especificada.

```
New-EC2VpnGateway -Type ipsec.1
```

### Salida:

```
AvailabilityZone :
State            : available
Tags             : {}
Type             : ipsec.1
VpcAttachments  : {}
VpnGatewayId    : vgw-1a2b3c4d
```

- Para obtener más información sobre la API, consulte [CreateVpnGateway](#) la referencia del AWS Tools for PowerShell cmdlet.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **DeleteCustomerGateway** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `DeleteCustomerGateway`.

## CLI

### AWS CLI

Para eliminar una pasarela de clientes

En este ejemplo, se elimina la pasarela de clientes especificada. Si el comando se ejecuta correctamente, no se muestra ningún resultado.

Comando:

```
aws ec2 delete-customer-gateway --customer-gateway-id cgw-0e11f167
```

- Para obtener más información sobre la API, consulte [DeleteCustomerGateway](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: en este ejemplo se elimina la pasarela de clientes especificada. Se le solicitará una confirmación antes de continuar con la operación, a menos que también especifique el parámetro Force.

```
Remove-EC2CustomerGateway -CustomerGatewayId cgw-1a2b3c4d
```

Salida:

```
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 obtener más información sobre la API, consulte [DeleteCustomerGateway](#) la referencia de AWS Tools for PowerShell cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo `DeleteDhcpOptions` con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `DeleteDhcpOptions`.

### CLI

#### AWS CLI

Para eliminar un conjunto de opciones de DHCP

En este ejemplo, se elimina el conjunto de opciones de DHCP especificado. Si el comando se ejecuta correctamente, no se muestra ningún resultado.

Comando:

```
aws ec2 delete-dhcp-options --dhcp-options-id dopt-d9070ebb
```

- Para obtener más información sobre la API, consulte la Referencia [DeleteDhcpOptions](#) de AWS CLI comandos.

### PowerShell

#### Herramientas para PowerShell

Ejemplo 1: en este ejemplo se elimina el conjunto de opciones de DHCP especificado. Se le solicitará una confirmación antes de continuar con la operación, a menos que también especifique el parámetro `Force`.

```
Remove-EC2DhcpOption -DhcpOptionsId dopt-1a2b3c4d
```

Salida:

```
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 obtener más información sobre la API, consulte [DeleteDhcpOptions](#) la referencia de AWS Tools for PowerShell cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **DeleteFlowLogs** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar DeleteFlowLogs.

### CLI

#### AWS CLI

Para eliminar un registro de flujo

En el siguiente delete-flow-logs ejemplo, se elimina el registro de flujo especificado.

```
aws ec2 delete-flow-logs --flow-log-id fl-11223344556677889
```

Salida:

```
{
  "Unsuccessful": []
}
```

- Para obtener más información sobre la API, consulte [DeleteFlowLogs](#) la Referencia de AWS CLI comandos.

### PowerShell

#### Herramientas para PowerShell

Ejemplo 1: Este ejemplo elimina el FlowLogId fl-01a2b3456a789c01 dado

```
Remove-EC2FlowLog -FlowLogId fl-01a2b3456a789c01
```

**Salida:**

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-EC2FlowLog (DeleteFlowLogs)" on target
"f1-01a2b3456a789c01".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
"Y"): Y
```

- Para obtener información sobre [DeleteFlowLogs AWS Tools for PowerShell](#) API, consulte la referencia del cmdlet.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **DeleteInternetGateway** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar DeleteInternetGateway.

### CLI

#### AWS CLI

Para eliminar una puerta de enlace a Internet

En el siguiente delete-internet-gateway ejemplo, se elimina la puerta de enlace de Internet especificada.

```
aws ec2 delete-internet-gateway \
  --internet-gateway-id igw-0d0fb496b3EXAMPLE
```

Este comando no genera ninguna salida.

Para obtener más información, consulte [Puertas de enlace de Internet](#) en la Guía del usuario de Amazon VPC.

- Para obtener más información sobre la API, consulte [DeleteInternetGateway](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: en este ejemplo se elimina la puerta de enlace de Internet especificada. Se le solicitará una confirmación antes de continuar con la operación, a menos que también especifique el parámetro Force.

```
Remove-EC2InternetGateway -InternetGatewayId igw-1a2b3c4d
```

#### Salida:

```
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 obtener más información sobre la API, consulte [DeleteInternetGateway](#) la referencia de AWS Tools for PowerShell cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **DeleteKeyPair** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar DeleteKeyPair.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en contexto en el siguiente ejemplo de código:

- [Comience a utilizar instancias](#)

## .NET

### AWS SDK for .NET

#### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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 (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 obtener más información sobre la API, consulta [DeleteKeyPair](#) la Referencia AWS SDK for .NET de la API.

## Bash

### AWS CLI con el script Bash

#### Note

Hay más información. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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
            ;;
        \?)
            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
}

```

Las funciones de utilidad utilizadas en este ejemplo.

```

#####
# 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 obtener más información sobre la API, consulta [DeleteKeyPair](#) la Referencia de AWS CLI comandos.

## C++

### SDK para C++

#### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

```
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;
}
}
```

- Para obtener más información sobre la API, consulta [DeleteKeyPair](#) la Referencia AWS SDK for C++ de la API.

## CLI

### AWS CLI

#### Eliminar un par de claves

En el siguiente `delete-key-pair` ejemplo, se elimina el key pair especificado.

```
aws ec2 delete-key-pair \
    --key-name my-key-pair
```

**Salida:**

```
{
  "Return": true,
  "KeyPairId": "key-03c8d3aceb53b507"
}
```

Para obtener más información, consulte [Crear y eliminar pares de claves](#) en la Guía del usuario de la interfaz de línea de AWS comandos.

- Para obtener más información sobre la API, consulte [DeleteKeyPair](#) la Referencia de AWS CLI comandos.

**Java****SDK para Java 2.x****Note**

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

```
public static void deleteKeys(Ec2Client ec2, String keyPair) {
    try {
        DeleteKeyPairRequest request = DeleteKeyPairRequest.builder()
            .keyName(keyPair)
            .build();

        ec2.deleteKeyPair(request);
        System.out.println("Successfully deleted key pair named " + keyPair);

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

- Para obtener más información sobre la API, consulta [DeleteKeyPair](#) la Referencia AWS SDK for Java 2.x de la API.

## JavaScript

### SDK para JavaScript (v3)

#### Note

Hay más información. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

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

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

export const main = async () => {
  const command = new DeleteKeyPairCommand({
    KeyName: "KEY_PAIR_NAME",
  });

  try {
    await client.send(command);
    console.log("Successfully deleted key pair.");
  } catch (err) {
    console.error(err);
  }
};
```

- Para obtener más información sobre la API, consulta [DeleteKeyPair](#) la Referencia AWS SDK for JavaScript de la API.

## Kotlin

### SDK para Kotlin

#### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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 obtener más información sobre la API, consulta [DeleteKeyPair](#) la referencia sobre el AWS SDK para la API de Kotlin.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: en este ejemplo se elimina el key pair especificado. Se le solicitará una confirmación antes de continuar con la operación, a menos que también especifique el parámetro Force.

```
Remove-EC2KeyPair -KeyName my-key-pair
```

Salida:

```
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 obtener más información sobre la API, consulte [DeleteKeyPair](#) la referencia de AWS Tools for PowerShell cmdlets.

## Python

### SDK para Python (Boto3)

#### Note

Hay más información al respecto. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

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

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

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

    def delete(self):
        """
        Deletes a key pair.
        """
        if self.key_pair is None:
```



```
        logger.info("No key pair to delete.")
        return

    key_name = self.key_pair.name
    try:
        self.key_pair.delete()
        self.key_pair = None
    except ClientError as err:
        logger.error(
            "Couldn't delete key %s. Here's why: %s : %s",
            key_name,
            err.response["Error"]["Code"],
            err.response["Error"]["Message"],
        )
        raise
```

- Para obtener más información sobre la API, consulta [DeleteKeyPair](#) la AWS Referencia de API de SDK for Python (Boto3).

## SAP ABAP

### SDK de SAP ABAP

#### Note

Hay más información al respecto. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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 obtener más información sobre la API, consulte [DeleteKeyPair](#) la referencia sobre la API ABAP del AWS SDK para SAP.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte [Cree recursos de Amazon EC2 mediante un SDK AWS](#). En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo `DeleteLaunchTemplate` con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `DeleteLaunchTemplate`.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en contexto en el siguiente ejemplo de código:

- [Creación y administración de un servicio resiliente](#)

.NET

AWS SDK for .NET

### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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 (AmazonClientException)  
    {  
        Console.WriteLine($"Unable to delete template {templateName}.");  
    }  
}
```

- Para obtener más información sobre la API, consulta [DeleteLaunchTemplate](#) la Referencia AWS SDK for .NET de la API.

## CLI

### AWS CLI

#### Eliminar una plantilla de lanzamiento

En este ejemplo, se elimina la plantilla de lanzamiento especificada.

Comando:

```
aws ec2 delete-launch-template --launch-template-id lt-0abcd290751193123
```

Salida:

```
{  
  "LaunchTemplate": {  
    "LatestVersionNumber": 2,  
    "LaunchTemplateId": "lt-0abcd290751193123",  
    "LaunchTemplateName": "TestTemplate",  
    "DefaultVersionNumber": 2,  
    "CreatedBy": "arn:aws:iam::123456789012:root",  
    "CreateTime": "2017-11-23T16:46:25.000Z"  
  }  
}
```

- Para obtener más información sobre la API, consulta [DeleteLaunchTemplate](#) la Referencia de AWS CLI comandos.

## JavaScript

### SDK para JavaScript (v3)

#### Note

Hay más información. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

```
await client.send(  
  new DeleteLaunchTemplateCommand({  
    LaunchTemplateName: NAMES.launchTemplateName,  
  }),  
);
```

- Para obtener más información sobre la API, consulta [DeleteLaunchTemplate](#) la Referencia AWS SDK for JavaScript de la API.

## Python

### SDK para Python (Boto3)

#### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

```
class AutoScaler:  
    """  
    Encapsulates Amazon EC2 Auto Scaling and EC2 management actions.  
    """  
  
    def __init__(  
        self,  
        resource_prefix,  
        inst_type,  
        ami_param,
```

```

        autoscaling_client,
        ec2_client,
        ssm_client,
        iam_client,
    ):
        """
        :param resource_prefix: The prefix for naming AWS resources that are
        created by this class.
        :param inst_type: The type of EC2 instance to create, such as t3.micro.
        :param ami_param: The Systems Manager parameter used to look up the AMI
        that is
            created.
        :param autoscaling_client: A Boto3 EC2 Auto Scaling client.
        :param ec2_client: A Boto3 EC2 client.
        :param ssm_client: A Boto3 Systems Manager client.
        :param iam_client: A Boto3 IAM client.
        """
        self.inst_type = inst_type
        self.ami_param = ami_param
        self.autoscaling_client = autoscaling_client
        self.ec2_client = ec2_client
        self.ssm_client = ssm_client
        self.iam_client = iam_client
        self.launch_template_name = f"{resource_prefix}-template"
        self.group_name = f"{resource_prefix}-group"
        self.instance_policy_name = f"{resource_prefix}-pol"
        self.instance_role_name = f"{resource_prefix}-role"
        self.instance_profile_name = f"{resource_prefix}-prof"
        self.bad_creds_policy_name = f"{resource_prefix}-bc-pol"
        self.bad_creds_role_name = f"{resource_prefix}-bc-role"
        self.bad_creds_profile_name = f"{resource_prefix}-bc-prof"
        self.key_pair_name = f"{resource_prefix}-key-pair"

    def delete_template(self):
        """
        Deletes a launch template.
        """
        try:
            self.ec2_client.delete_launch_template(
                LaunchTemplateName=self.launch_template_name
            )
            self.delete_instance_profile(
                self.instance_profile_name, self.instance_role_name
            )

```

```
    )
    log.info("Launch template %s deleted.", self.launch_template_name)
except ClientError as err:
    if (
        err.response["Error"]["Code"]
        == "InvalidLaunchTemplateName.NotFoundException"
    ):
        log.info(
            "Launch template %s does not exist, nothing to do.",
            self.launch_template_name,
        )
    else:
        raise AutoScalerError(
            f"Couldn't delete launch template
{self.launch_template_name}: {err}."
        )
```

- Para obtener más información sobre la API, consulta [DeleteLaunchTemplate](#) la AWS Referencia de API de SDK for Python (Boto3).

Para obtener una lista completa de las guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **DeleteNetworkAcl** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `DeleteNetworkAcl`.

### CLI

#### AWS CLI

Para eliminar una ACL de red

En este ejemplo, se elimina la ACL de red especificada. Si el comando se ejecuta correctamente, no se muestra ningún resultado.

Comando:

```
aws ec2 delete-network-acl --network-acl-id acl-5fb85d36
```

- Para obtener información sobre la API, consulte [DeleteNetworkAcl](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: En este ejemplo se elimina la ACL de red especificada. Se le solicitará una confirmación antes de continuar con la operación, a menos que también especifique el parámetro Force.

```
Remove-EC2NetworkAcl -NetworkAclId acl-12345678
```

### Salida:

```
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 obtener más información sobre la API, consulte [DeleteNetworkAcl](#) la referencia de AWS Tools for PowerShell cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **DeleteNetworkAclEntry** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `DeleteNetworkAclEntry`.

### CLI

#### AWS CLI

Para eliminar una entrada de ACL de red

En este ejemplo, se elimina la regla de entrada número 100 de la ACL de red especificada. Si el comando se ejecuta correctamente, no se muestra ningún resultado.

Comando:

```
aws ec2 delete-network-acl-entry --network-acl-id acl-5fb85d36 --ingress --rule-number 100
```

- Para obtener más información sobre la API, consulte la Referencia [DeleteNetworkAclEntry](#) de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: Este ejemplo elimina la regla especificada de la ACL de red especificada. Se le solicitará la confirmación antes de continuar con la operación, a menos que también especifique el parámetro Force.

```
Remove-EC2NetworkAclEntry -NetworkAclId acl-12345678 -Egress $false -RuleNumber 100
```

Salida:

```
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 obtener más información sobre la API, consulte [DeleteNetworkAclEntry](#) la referencia de AWS Tools for PowerShell cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.



## Úselo `DeleteNetworkInterface` con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `DeleteNetworkInterface`.

### CLI

#### AWS CLI

Para eliminar una interfaz de red

En este ejemplo, se elimina la interfaz de red especificada. Si el comando se ejecuta correctamente, no se muestra ningún resultado.

Comando:

```
aws ec2 delete-network-interface --network-interface-id eni-e5aa89a3
```

- Para obtener más información sobre la API, consulte [DeleteNetworkInterface](#) la Referencia de AWS CLI comandos.

### PowerShell

#### Herramientas para PowerShell

Ejemplo 1: en este ejemplo se elimina la interfaz de red especificada. Se le solicitará una confirmación antes de continuar con la operación, a menos que también especifique el parámetro `Force`.

```
Remove-EC2NetworkInterface -NetworkInterfaceId eni-12345678
```

Salida:

```
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 obtener más información sobre la API, consulte [DeleteNetworkInterface](#) la referencia de AWS Tools for PowerShell cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **DeletePlacementGroup** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `DeletePlacementGroup`.

### CLI

#### AWS CLI

Para eliminar un grupo de ubicaciones

Este comando de ejemplo elimina el grupo de ubicación especificado.

Comando:

```
aws ec2 delete-placement-group --group-name my-cluster
```

- Para obtener más información sobre la API, consulte [DeletePlacementGroup](#) la Referencia de AWS CLI comandos.

### PowerShell

#### Herramientas para PowerShell

Ejemplo 1: en este ejemplo se elimina el grupo de ubicación especificado. Se le solicitará una confirmación antes de continuar con la operación, a menos que también especifique el parámetro `Force`.

```
Remove-EC2PlacementGroup -GroupName my-placement-group
```

Salida:

```
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 obtener más información sobre la API, consulte [DeletePlacementGroup](#) la referencia de AWS Tools for PowerShell cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **DeleteRoute** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `DeleteRoute`.

### CLI

#### AWS CLI

Para eliminar una ruta

En este ejemplo, se elimina la ruta especificada de la tabla de rutas especificada. Si el comando se ejecuta correctamente, no se muestra ningún resultado.

Comando:

```
aws ec2 delete-route --route-table-id rtb-22574640 --destination-cidr-block 0.0.0.0/0
```

- Para obtener más información sobre la API, consulte [DeleteRoute](#) la Referencia de AWS CLI comandos.

### PowerShell

#### Herramientas para PowerShell

Ejemplo 1: en este ejemplo se elimina la ruta especificada de la tabla de rutas especificada. Se le solicitará la confirmación antes de continuar con la operación, a menos que también especifique el parámetro `Force`.

```
Remove-EC2Route -RouteTableId rtb-1a2b3c4d -DestinationCidrBlock 0.0.0.0/0
```

Salida:

**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 obtener más información sobre la API, consulte [DeleteRoute](#) la referencia de AWS Tools for PowerShell cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **DeleteRouteTable** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar DeleteRouteTable.

### CLI

#### AWS CLI

Para eliminar una tabla de rutas

En este ejemplo, se elimina la tabla de rutas especificada. Si el comando se ejecuta correctamente, no se muestra ningún resultado.

Comando:

```
aws ec2 delete-route-table --route-table-id rtb-22574640
```

- Para obtener detalles sobre la API, consulte [DeleteRouteTable](#) la Referencia de AWS CLI comandos.

### PowerShell

#### Herramientas para PowerShell

Ejemplo 1: en este ejemplo se elimina la tabla de rutas especificada. Se le solicitará una confirmación antes de continuar con la operación, a menos que también especifique el parámetro Force.

```
Remove-EC2RouteTable -RouteTableId rtb-1a2b3c4d
```

**Salida:**

```
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 obtener más información sobre la API, consulte [DeleteRouteTable](#) la referencia de AWS Tools for PowerShell cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **DeleteSecurityGroup** con un AWS SDK o CLI


En los siguientes ejemplos de código, se muestra cómo utilizar `DeleteSecurityGroup`.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en contexto en el siguiente ejemplo de código:

- [Comience a utilizar instancias](#)

### .NET

#### AWS SDK for .NET

** Note**

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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)
    {
        var response = await _amazonEC2.DeleteSecurityGroupAsync(new
DeleteSecurityGroupRequest { GroupId = groupId });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }

```

- Para obtener más información sobre la API, consulta [DeleteSecurityGroup](#) la Referencia AWS SDK for .NET de la API.

## Bash

### AWS CLI con el script Bash

#### Note

Hay más información. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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
}

```

Las funciones de utilidad utilizadas en este ejemplo.

```
#####  
# 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 obtener más información sobre la API, consulta [DeleteSecurityGroup](#) la Referencia de AWS CLI comandos.

## C++

### SDK para C++

#### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

```
Aws::EC2::EC2Client ec2Client(clientConfiguration);
Aws::EC2::Model::DeleteSecurityGroupRequest request;

request.SetGroupId(securityGroupID);
auto outcome = ec2Client.DeleteSecurityGroup(request);

if (!outcome.IsSuccess()) {
    std::cerr << "Failed to delete security group " << securityGroupID <<
        ":" << outcome.GetError().GetMessage() << std::endl;
}
else {
    std::cout << "Successfully deleted security group " << securityGroupID <<
        std::endl;
}
```

- Para obtener más información sobre la API, consulta [DeleteSecurityGroup](#) la Referencia AWS SDK for C++ de la API.

## CLI

### AWS CLI

#### [EC2-Classical] Eliminar un grupo de seguridad

En este ejemplo, se elimina el grupo de seguridad denominado MySecurityGroup. Si el comando se ejecuta correctamente, no se muestra ningún resultado.

Comando:

```
aws ec2 delete-security-group --group-name MySecurityGroup
```

### [EC2-VPC] Eliminar un grupo de seguridad

En este ejemplo, se elimina el grupo de seguridad con el ID `sg-903004f8`. Tenga en cuenta que no puede hacer referencia a un grupo de seguridad de EC2-VPC por nombre. Si el comando se ejecuta correctamente, no se muestra ningún resultado.

Comando:

```
aws ec2 delete-security-group --group-id sg-903004f8
```

Para obtener más información, consulte [Uso de los grupos de seguridad](#) en la Guía del usuario de la Interfaz de la línea de comandos de AWS .

- Para obtener más información sobre la API, consulta [DeleteSecurityGroup](#) la Referencia de AWS CLI comandos.

## Java

### SDK para Java 2.x

#### Note

Hay más información al respecto [en GitHub](#). Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

```
public static void deleteEC2SecGroup(Ec2Client ec2, String groupId) {
    try {
        DeleteSecurityGroupRequest request =
DeleteSecurityGroupRequest.builder()
        .groupId(groupId)
        .build();

        ec2.deleteSecurityGroup(request);
        System.out.println("Successfully deleted security group with Id " +
groupId);
    }
}
```

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

- Para obtener más información sobre la API, consulta [DeleteSecurityGroup](#) la Referencia AWS SDK for Java 2.x de la API.

## JavaScript

### SDK para JavaScript (v3)

#### Note

Hay más información. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

```
import { DeleteSecurityGroupCommand } from "@aws-sdk/client-ec2";  
  
import { client } from "../libs/client.js";  
  
export const main = async () => {  
    const command = new DeleteSecurityGroupCommand({  
        GroupId: "GROUP_ID",  
    });  
  
    try {  
        await client.send(command);  
        console.log("Security group deleted successfully.");  
    } catch (err) {  
        console.error(err);  
    }  
};
```

- Para obtener más información sobre la API, consulta [DeleteSecurityGroup](#) la Referencia AWS SDK for JavaScript de la API.

## Kotlin

### SDK para Kotlin

#### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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 obtener más información sobre la API, consulta [DeleteSecurityGroup](#) la referencia sobre el AWS SDK para la API de Kotlin.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: en este ejemplo se elimina el grupo de seguridad especificado para EC2-VPC. Se le solicitará una confirmación antes de continuar con la operación, a menos que también especifique el parámetro Force.

```
Remove-EC2SecurityGroup -GroupId sg-12345678
```

#### Salida:

```
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"):
```

Ejemplo 2: en este ejemplo se elimina el grupo de seguridad especificado para EC2-Classic.

```
Remove-EC2SecurityGroup -GroupName my-security-group -Force
```

- Para obtener más información sobre la API, consulte la referencia de [DeleteSecurityGroup](#) cmdlets AWS Tools for PowerShell .

## Python

### SDK para Python (Boto3)

#### Note

Hay más información al respecto. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

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

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

    @classmethod
    def from_resource(cls):
```

```
ec2_resource = boto3.resource("ec2")
return cls(ec2_resource)

def delete(self):
    """
    Deletes the security group.
    """
    if self.security_group is None:
        logger.info("No security group to delete.")
        return

    group_id = self.security_group.id
    try:
        self.security_group.delete()
    except ClientError as err:
        logger.error(
            "Couldn't delete security group %s. Here's why: %s: %s",
            group_id,
            err.response["Error"]["Code"],
            err.response["Error"]["Message"],
        )
        raise
```

- Para obtener más información sobre la API, consulta [DeleteSecurityGroup](#) la AWS Referencia de API de SDK for Python (Boto3).

## SAP ABAP

### SDK de SAP ABAP

#### Note

Hay más información al respecto. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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 obtener más información sobre la API, consulte [DeleteSecurityGroup](#) la referencia sobre la API ABAP del AWS SDK para SAP.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte [Cree recursos de Amazon EC2 mediante un SDK AWS](#). En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **DeleteSnapshot** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar DeleteSnapshot.

### CLI

#### AWS CLI

##### Eliminar una instantánea

Este comando de ejemplo elimina la instantánea con el ID de instantánea de snap-1234567890abcdef0. Si el comando se ejecuta correctamente, no se muestra ningún resultado.

Comando:

```
aws ec2 delete-snapshot --snapshot-id snap-1234567890abcdef0
```

- Para obtener más información sobre la API, consulte [DeleteSnapshot](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: en este ejemplo se elimina la instantánea especificada. Se le solicitará una confirmación antes de continuar con la operación, a menos que también especifique el parámetro Force.

```
Remove-EC2Snapshot -SnapshotId snap-12345678
```

Salida:

```
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 obtener más información sobre la API, consulte [DeleteSnapshot](#) la referencia de AWS Tools for PowerShell cmdlets.

## Rust

### SDK para Rust

#### Note

Hay más información al respecto. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

```
async fn delete_snapshot(client: &Client, id: &str) -> Result<(), Error> {
    client.delete_snapshot().snapshot_id(id).send().await?;

    println!("Deleted");

    Ok(())
}
```



- Para obtener más información sobre la API, consulta [DeleteSnapshot](#) la referencia sobre la API de AWS SDK para Rust.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte [Cree recursos de Amazon EC2 mediante un SDK AWS](#). En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **DeleteSpotDatafeedSubscription** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `DeleteSpotDatafeedSubscription`.

### CLI

#### AWS CLI

Para cancelar una suscripción a una fuente de datos de Spot Instance

Este comando de ejemplo elimina una suscripción a una fuente de datos de Spot de la cuenta. Si el comando se ejecuta correctamente, no se muestra ningún resultado.

Comando:

```
aws ec2 delete-spot-datafeed-subscription
```

- Para obtener más información sobre la API, consulte [DeleteSpotDatafeedSubscription](#) la Referencia de AWS CLI comandos.

### PowerShell

#### Herramientas para PowerShell

Ejemplo 1: en este ejemplo se elimina la fuente de datos de la instancia de Spot. Se le solicitará una confirmación antes de continuar con la operación, a menos que también especifique el parámetro `Force`.

```
Remove-EC2SpotDatafeedSubscription
```

Salida:

```
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 obtener más información sobre la API, consulte [DeleteSpotDatafeedSubscription](#) la referencia de AWS Tools for PowerShell cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **DeleteSubnet** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `DeleteSubnet`.

### CLI

#### AWS CLI

Para eliminar una subred

En este ejemplo, se elimina la subred especificada. Si el comando se ejecuta correctamente, no se muestra ningún resultado.

Comando:

```
aws ec2 delete-subnet --subnet-id subnet-9d4a7b6c
```

- Para obtener más información sobre la API, consulte la Referencia [DeleteSubnet](#) de AWS CLI comandos.

### PowerShell

#### Herramientas para PowerShell

Ejemplo 1: en este ejemplo se elimina la subred especificada. Se le solicitará una confirmación antes de continuar con la operación, a menos que también especifique el parámetro `Force`.

```
Remove-EC2Subnet -SubnetId subnet-1a2b3c4d
```

### Salida:

```
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 obtener más información sobre la API, consulte [DeleteSubnet](#) la referencia de AWS Tools for PowerShell cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **DeleteTags** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar DeleteTags.

### CLI

#### AWS CLI

Ejemplo 1: Para eliminar una etiqueta de un recurso

En el siguiente `delete-tags` ejemplo, se elimina la etiqueta `Stack=Test` de la imagen especificada. Al especificar un valor y un nombre de clave, la etiqueta se elimina solo si el valor de la etiqueta coincide con el valor especificado.

```
aws ec2 delete-tags \  
  --resources ami-1234567890abcdef0 \  
  --tags Key=Stack,Value=Test
```

Es opcional especificar el valor de una etiqueta. En el siguiente `delete-tags` ejemplo, se elimina la etiqueta con el nombre `purpose` de clave de la instancia especificada, independientemente del valor de la etiqueta.

```
aws ec2 delete-tags \  
  --resources i-1234567890abcdef0 \  
  --tags Key=purpose
```

Si especifica la cadena vacía como el valor de la etiqueta, la etiqueta se eliminará solo si el valor de la etiqueta es la cadena vacía. En el siguiente `delete-tags` ejemplo, se especifica la cadena vacía como valor de etiqueta de la etiqueta que se va a eliminar.

```
aws ec2 delete-tags \  
  --resources i-1234567890abcdef0 \  
  --tags Key=Name,Value=
```

Ejemplo 2: Para eliminar una etiqueta de varios recursos

En el siguiente `delete-tags` ejemplo, se elimina la etiqueta ```Purpose=Test``` tanto de una instancia como de una AMI. Como se muestra en el ejemplo anterior, puedes omitir el valor de la etiqueta en el comando.

```
aws ec2 delete-tags \  
  --resources i-1234567890abcdef0 ami-1234567890abcdef0 \  
  --tags Key=Purpose
```

- Para obtener más información sobre la API, consulte [DeleteTags](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: en este ejemplo se elimina la etiqueta especificada del recurso especificado, independientemente del valor de la etiqueta. La sintaxis utilizada en este ejemplo requiere la PowerShell versión 3 o posterior.

```
Remove-EC2Tag -Resource i-12345678 -Tag @{ Key="myTag" } -Force
```

Ejemplo 2: en este ejemplo se elimina la etiqueta especificada del recurso especificado, pero solo si el valor de la etiqueta coincide. La sintaxis utilizada en este ejemplo requiere la PowerShell versión 3 o posterior.

```
Remove-EC2Tag -Resource i-12345678 -Tag @{ Key="myTag";Value="myTagValue" } -Force
```

Ejemplo 3: en este ejemplo se elimina la etiqueta especificada del recurso especificado, independientemente del valor de la etiqueta.

```
$tag = New-Object Amazon.EC2.Model.Tag
$tag.Key = "myTag"

Remove-EC2Tag -Resource i-12345678 -Tag $tag -Force
```

Ejemplo 4: en este ejemplo se elimina la etiqueta especificada del recurso especificado, pero solo si el valor de la etiqueta coincide.

```
$tag = New-Object Amazon.EC2.Model.Tag
$tag.Key = "myTag"
$tag.Value = "myTagValue"

Remove-EC2Tag -Resource i-12345678 -Tag $tag -Force
```

- Para obtener más información sobre la API, consulte la referencia [DeleteTags](#) del AWS Tools for PowerShell cmdlet.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **DeleteVolume** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `DeleteVolume`.

### CLI

#### AWS CLI

Para eliminar un volumen

Este comando de ejemplo elimina un volumen disponible con el identificador de volumen `devol-049df61146c4d7901`. Si el comando se ejecuta correctamente, no se muestra ningún resultado.

## Comando:

```
aws ec2 delete-volume --volume-id vol-049df61146c4d7901
```

- Para obtener más información sobre la API, consulte [DeleteVolume](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: En este ejemplo se separa el volumen especificado. Se le solicitará una confirmación antes de continuar con la operación, a menos que también especifique el parámetro Force.

```
Remove-EC2Volume -VolumeId vol-12345678
```

## Salida:

```
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 obtener más información sobre la API, consulte [DeleteVolume](#) la referencia de AWS Tools for PowerShell cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **DeleteVpc** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `DeleteVpc`.

## CLI

### AWS CLI

#### Para eliminar una VPC

En este ejemplo, se elimina la VPC especificada. Si el comando se ejecuta correctamente, no se muestra ningún resultado.

Comando:

```
aws ec2 delete-vpc --vpc-id vpc-a01106c2
```

- Para obtener información sobre la API, consulte la Referencia [DeleteVpc](#) de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: en este ejemplo se elimina la VPC especificada. Se le solicitará una confirmación antes de continuar con la operación, a menos que también especifique el parámetro Force.

```
Remove-EC2Vpc -VpcId vpc-12345678
```

Salida:

```
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 obtener más información sobre la API, consulte [DeleteVpc](#) la referencia de AWS Tools for PowerShell cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo `DeleteVpnConnection` con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `DeleteVpnConnection`.

### CLI

#### AWS CLI

Para eliminar una conexión VPN

En este ejemplo, se elimina la conexión VPN especificada. Si el comando se ejecuta correctamente, no se muestra ningún resultado.

Comando:

```
aws ec2 delete-vpn-connection --vpn-connection-id vpn-40f41529
```

- Para obtener más información sobre la API, consulte [DeleteVpnConnection](#) la Referencia de AWS CLI comandos.

### PowerShell

#### Herramientas para PowerShell

Ejemplo 1: en este ejemplo se elimina la conexión VPN especificada. Se le solicitará una confirmación antes de continuar con la operación, a menos que también especifique el parámetro `Force`.

```
Remove-EC2VpnConnection -VpnConnectionId vpn-12345678
```

Salida:

```
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 obtener más información sobre la API, consulte [DeleteVpnConnection](#) la referencia de AWS Tools for PowerShell cmdlets.



Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo `DeleteVpnConnectionRoute` con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `DeleteVpnConnectionRoute`.

### CLI

#### AWS CLI

Para eliminar una ruta estática de una conexión VPN

En este ejemplo, se elimina la ruta estática especificada de la conexión VPN especificada. Si el comando se ejecuta correctamente, no se muestra ningún resultado.

Comando:

```
aws ec2 delete-vpn-connection-route --vpn-connection-id vpn-40f41529 --
destination-cidr-block 11.12.0.0/16
```

- Para obtener más información sobre la API, consulte [DeleteVpnConnectionRoute](#) la Referencia de AWS CLI comandos.

### PowerShell

#### Herramientas para PowerShell

Ejemplo 1: este ejemplo elimina la ruta estática especificada de la conexión VPN especificada. Se le solicitará la confirmación antes de continuar con la operación, a menos que también especifique el parámetro `Force`.

```
Remove-EC2VpnConnectionRoute -VpnConnectionId vpn-12345678 -DestinationCidrBlock
11.12.0.0/16
```

Salida:

```
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 obtener más información sobre la API, consulte [DeleteVpnConnectionRoute](#) la referencia de AWS Tools for PowerShell cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **DeleteVpnGateway** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `DeleteVpnGateway`.

### CLI

#### AWS CLI

Para eliminar una puerta de enlace privada virtual

En este ejemplo, se elimina la puerta de enlace privada virtual especificada. Si el comando se ejecuta correctamente, no se muestra ningún resultado.

Comando:

```
aws ec2 delete-vpn-gateway --vpn-gateway-id vgw-9a4cacf3
```

- Para obtener más información sobre la API, consulte [DeleteVpnGateway](#) la Referencia de AWS CLI comandos.

### PowerShell

#### Herramientas para PowerShell

Ejemplo 1: en este ejemplo se elimina la puerta de enlace privada virtual especificada. Se le solicitará una confirmación antes de continuar con la operación, a menos que también especifique el parámetro `Force`.

```
Remove-EC2VpnGateway -VpnGatewayId vgw-1a2b3c4d
```

Salida:

```
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 obtener más información sobre la API, consulte [DeleteVpnGateway](#) la referencia de AWS Tools for PowerShell cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **DeregisterImage** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `DeregisterImage`.

CLI

AWS CLI

Para anular el registro de una AMI

En este ejemplo se anula el registro de la AMI especificada. Si el comando se ejecuta correctamente, no se muestra ningún resultado.

Comando:

```
aws ec2 deregister-image --image-id ami-4fa54026
```

- Para obtener más información sobre la API, consulte la Referencia [DeregisterImage](#) de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: En este ejemplo se anula el registro de la AMI especificada.

```
Unregister-EC2Image -ImageId ami-12345678
```

- Para obtener más información sobre la API, consulte la referencia del [DeregisterImage AWS Tools for PowerShell](#) cmdlet.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **DescribeAccountAttributes** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar DescribeAccountAttributes.

### CLI

#### AWS CLI

Para describir todos los atributos de tu AWS cuenta

En este ejemplo se describen los atributos de su AWS cuenta.

Comando:

```
aws ec2 describe-account-attributes
```

Salida:

```
{
  "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 describir un único atributo de su AWS cuenta

En este ejemplo se describe el `supported-platforms` atributo de tu AWS cuenta.

Comando:

```
aws ec2 describe-account-attributes --attribute-names supported-platforms
```

Salida:

```
{
  "AccountAttributes": [
    {
      "AttributeName": "supported-platforms",
      "AttributeValues": [
        {
          "AttributeValue": "EC2"
        },
        {
          "AttributeValue": "VPC"
        }
      ]
    }
  ]
}
```

- Para obtener más información sobre la API, consulta [DescribeAccountAttributes](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: en este ejemplo se describe si puede lanzar instancias en EC2-Classic y EC2-VPC en la región, o solo en EC2-VPC.

```
(Get-EC2AccountAttribute -AttributeName supported-platforms).AttributeValues
```

Salida:

```
AttributeValue
-----
EC2
VPC
```

Ejemplo 2: Este ejemplo describe su VPC predeterminada o es «ninguna» si no tiene una VPC predeterminada en la región.

```
(Get-EC2AccountAttribute -AttributeName default-vpc).AttributeValues
```

Salida:

```
AttributeValue
-----
vpc-12345678
```

Ejemplo 3: en este ejemplo se describe la cantidad máxima de instancias bajo demanda que puede ejecutar.

```
(Get-EC2AccountAttribute -AttributeName max-instances).AttributeValues
```

Salida:

```
AttributeValue
-----
20
```

- Para obtener más información sobre la API, consulte [DescribeAccountAttributes](#) la referencia de AWS Tools for PowerShell cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **DescribeAddresses** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar DescribeAddresses.

## C++

## SDK para C++

 Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

```
Aws::EC2::EC2Client ec2Client(clientConfiguration);
Aws::EC2::Model::DescribeAddressesRequest request;
auto outcome = ec2Client.DescribeAddresses(request);
if (outcome.IsSuccess()) {
    std::cout << std::left << std::setw(20) << "InstanceId" <<
                std::setw(15) << "Public IP" << std::setw(10) << "Domain" <<
                std::setw(30) << "Allocation ID" << std::setw(25) <<
                "NIC ID" << std::endl;

    const auto &addresses = outcome.GetResult().GetAddresses();
    for (const auto &address: addresses) {
        Aws::String domainString =
            Aws::EC2::Model::DomainTypeMapper::GetNameForDomainType(
                address.GetDomain());

        std::cout << std::left << std::setw(20) <<
                    address.GetInstanceId() << std::setw(15) <<
                    address.GetPublicIp() << std::setw(10) << domainString <<
                    std::setw(30) << address.GetAllocationId() << std::setw(25)
                    << address.GetNetworkInterfaceId() << std::endl;
    }
}
else {
    std::cerr << "Failed to describe Elastic IP addresses:" <<
                outcome.GetError().GetMessage() << std::endl;
}
```

- Para obtener más información sobre la API, consulta [DescribeAddresses](#) la Referencia AWS SDK for C++ de la API.



## CLI

## AWS CLI

## Ejemplo 1: Recuperar detalles sobre todas las direcciones IP elásticas

En el siguiente ejemplo de `describe-addresses`, se muestran los detalles de las direcciones IP elásticas.

```
aws ec2 describe-addresses
```

Salida:

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

## Ejemplo 2: Recuperar detalles sobre las direcciones IP elásticas de EC2-VPC

En el siguiente ejemplo de `describe-addresses`, se muestran detalles sobre las direcciones IP elásticas para usarlas con las instancias de una VPC.

```
aws ec2 describe-addresses \
```

```
--filters "Name=domain,Values=vpc"
```

Salida:

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

Ejemplo 3: Recuperar detalles sobre una dirección IP elástica especificada por el ID de asignación

En el siguiente ejemplo de `describe-addresses`, se muestran los detalles sobre la dirección IP elástica con el ID de asignación especificado, el cual está asociado a una instancia en EC2-VPC.

```
aws ec2 describe-addresses \
  --allocation-ids eipalloc-282d9641
```

Salida:

```
{
  "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"
    }
]
}

```

**Ejemplo 4:** Recuperar detalles sobre una dirección IP elástica especificada por la dirección IP privada de su VPC

En el siguiente ejemplo de `describe-addresses`, se muestran los detalles sobre la dirección IP elástica asociada a una dirección IP privada en particular en EC2-VPC.

```

aws ec2 describe-addresses \
  --filters "Name=private-ip-address,Values=10.251.50.12"

```

**Ejemplo 5:** Recuperar detalles sobre las direcciones IP elásticas en EC2-Classic

En el siguiente ejemplo de `describe-addresses`, se muestran los detalles de una dirección IP elástica para su uso en EC2-Classic.

```

aws ec2 describe-addresses \
  --filters "Name=domain,Values=standard"

```

**Salida:**

```

{
  "Addresses": [
    {
      "InstanceId": "i-1234567890abcdef0",
      "PublicIp": "203.0.110.25",
      "PublicIpv4Pool": "amazon",
      "Domain": "standard"
    }
  ]
}

```

**Ejemplo 6:** Recuperar detalles sobre una dirección IP elástica especificada por la dirección IP pública

En el siguiente ejemplo de `describe-addresses`, se muestran detalles sobre la dirección IP elástica con el valor `203.0.110.25`, el cual está asociado a una instancia en EC2-Classic.

```
aws ec2 describe-addresses \  
  --public-ips 203.0.110.25
```

Salida:

```
{  
  "Addresses": [  
    {  
      "InstanceId": "i-1234567890abcdef0",  
      "PublicIp": "203.0.110.25",  
      "PublicIpv4Pool": "amazon",  
      "Domain": "standard"  
    }  
  ]  
}
```

- Para obtener más información sobre la API, consulta [DescribeAddresses](#) la Referencia de AWS CLI comandos.

## JavaScript

### SDK para JavaScript (v3)

#### Note

Hay más información. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

```
import { DescribeAddressesCommand } from "@aws-sdk/client-ec2";  
  
import { client } from "../libs/client.js";  
  
export const main = async () => {  
  const command = new DescribeAddressesCommand({  
    // You can omit this property to show all addresses.  
    AllocationIds: ["ALLOCATION_ID"],  
  });  
  
  try {
```

```
const { Addresses } = await client.send(command);
const addressList = Addresses.map((address) => ` • ${address.PublicIp}`);
console.log("Elastic IP addresses:");
console.log(addressList.join("\n"));
} catch (err) {
  console.error(err);
}
};
```

- Para obtener más información sobre la API, consulta [DescribeAddresses](#) la Referencia AWS SDK for JavaScript de la API.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: en este ejemplo se describe la dirección IP elástica especificada para las instancias de EC2-Classic.

```
Get-EC2Address -AllocationId eipalloc-12345678
```

Salida:

```
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
```

Ejemplo 2: en este ejemplo se describen las direcciones IP elásticas para las instancias de una VPC. Esta sintaxis requiere la PowerShell versión 3 o posterior.

```
Get-EC2Address -Filter @{ Name="domain";Values="vpc" }
```

Ejemplo 3: en este ejemplo se describe la dirección IP elástica especificada para las instancias de EC2-Classic.

```
Get-EC2Address -PublicIp 203.0.113.17
```

**Salida:**

```
AllocationId      :
AssociationId     :
Domain           : standard
InstanceId        : i-12345678
NetworkInterfaceId :
NetworkInterfaceOwnerId :
PrivateIpAddress  :
PublicIp         : 203.0.113.17
```

Ejemplo 4: en este ejemplo se describen las direcciones IP elásticas para las instancias de EC2-Classic. Esta sintaxis requiere la PowerShell versión 3 o posterior.

```
Get-EC2Address -Filter @{ Name="domain";Values="standard" }
```

Ejemplo 5: en este ejemplo se describen todas las direcciones IP elásticas.

```
Get-EC2Address
```

Ejemplo 6: Este ejemplo devuelve la IP pública y privada del identificador de instancia proporcionado en el filtro

```
Get-EC2Address -Region eu-west-1 -Filter @{Name="instance-id";Values="i-0c12d3f4f567ffb89"} | Select-Object PrivateIpAddress, PublicIp
```

**Salida:**

```
PrivateIpAddress PublicIp
-----
10.0.0.99         63.36.5.227
```

Ejemplo 7: Este ejemplo recupera todas las IP elásticas con su identificador de asignación, identificador de asociación e identificador de instancia

```
Get-EC2Address -Region eu-west-1 | Select-Object InstanceId, AssociationId, AllocationId, PublicIp
```

**Salida:**

InstanceId	AssociationId	AllocationId
PublicIp		
-----	-----	-----
-----		
		eipalloc-012e3b456789e1fad
17.212.120.178		
i-0c123dfd3415bac67	eipassoc-0e123456bb7890bdb	eipalloc-01cd23ebf45f7890c
17.212.124.77		
		eipalloc-012345678eeabcfad
17.212.225.7		
i-0123d405c67e89a0c	eipassoc-0c123b456783966ba	eipalloc-0123cdd456a8f7892
37.216.52.173		
i-0f1bf2f34c5678d09	eipassoc-0e12934568a952d96	eipalloc-0e1c23e4d5e6789e4
37.218.222.278		
i-012e3cb4df567e8aa	eipassoc-0d1b2fa4d67d03810	eipalloc-0123f456f78a01b58
37.210.82.27		
i-0123bcf4b567890e1	eipassoc-01d2345f678903fb1	eipalloc-0e1db23cfef5c45c7
37.215.222.270		

**Ejemplo 8:** En este ejemplo se obtiene una lista de direcciones IP de EC2 que coinciden con la clave de etiqueta «Categoría» con el valor «Prod»

```
Get-EC2Address -Filter @{Name="tag:Category";Values="Prod"}
```

**Salida:**

```
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 obtener más información sobre la API, consulte la referencia del [DescribeAddresses](#) cmdlet AWS Tools for PowerShell .

## SAP ABAP

### SDK de SAP ABAP

#### Note

Hay más información al respecto. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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 obtener más información sobre la API, consulte [DescribeAddresses](#) la referencia sobre la API ABAP del AWS SDK para SAP.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte [Cree recursos de Amazon EC2 mediante un SDK AWS](#). En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **DescribeAvailabilityZones** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar DescribeAvailabilityZones.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en contexto en el siguiente ejemplo de código:

- [Creación y administración de un servicio resiliente](#)



## .NET

### AWS SDK for .NET

#### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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()
{
    var zoneResponse = await _amazonEc2.DescribeAvailabilityZonesAsync(
        new DescribeAvailabilityZonesRequest());
    return zoneResponse.AvailabilityZones.Select(z => z.ZoneName).ToList();
}
```

- Para obtener más información sobre la API, consulta [DescribeAvailabilityZones](#) la Referencia AWS SDK for .NET de la API.

## C++

### SDK para C++

#### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

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

Aws::EC2::Model::DescribeAvailabilityZonesRequest describe_request;
```

```
auto describe_outcome =
ec2Client.DescribeAvailabilityZones(describe_request);

if (describe_outcome.IsSuccess()) {
    std::cout << std::left <<
        std::setw(32) << "ZoneName" <<
        std::setw(20) << "State" <<
        std::setw(32) << "Region" << std::endl;

    const auto &zones =
        describe_outcome.GetResult().GetAvailabilityZones();

    for (const auto &zone: zones) {
        Aws::String stateString =
            Aws::EC2::Model::AvailabilityZoneStateMapper::GetNameForAvailabilityZoneState(
                zone.GetState());
        std::cout << std::left <<
            std::setw(32) << zone.GetZoneName() <<
            std::setw(20) << stateString <<
            std::setw(32) << zone.GetRegionName() << std::endl;
    }
}
else {
    std::cerr << "Failed to describe availability zones:" <<
        describe_outcome.GetError().GetMessage() << std::endl;
    result = false;
}
```

- Para obtener más información sobre la API, consulta [DescribeAvailabilityZones](#) la Referencia AWS SDK for C++ de la API.

## CLI

### AWS CLI

#### Describir las zonas de disponibilidad

En el siguiente ejemplo de `describe-availability-zones`, se muestran los detalles de las zonas de disponibilidad que están disponibles para usted. La respuesta incluye las zonas

de disponibilidad solo para la región actual. En este ejemplo, se usa la región predeterminada del perfil us-west-2 (Oregón).

```
aws ec2 describe-availability-zones
```

Salida:

```
{
  "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 obtener más información sobre la API, consulta [DescribeAvailabilityZones](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: En este ejemplo se describen las zonas de disponibilidad de la región actual que están disponibles para usted.

```
Get-EC2AvailabilityZone
```

Salida:

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

Ejemplo 2: Este ejemplo describe cualquier zona de disponibilidad que se encuentre en un estado deteriorado. La sintaxis utilizada en este ejemplo requiere PowerShell la versión 3 o superior.

```
Get-EC2AvailabilityZone -Filter @{ Name="state";Values="impaired" }
```

Ejemplo 3: Con la PowerShell versión 2, debe usar New-Object para crear el filtro.

```
$filter = New-Object Amazon.EC2.Model.Filter
$filter.Name = "state"
$filter.Values = "impaired"

Get-EC2AvailabilityZone -Filter $filter
```

- Para obtener más información sobre la API, consulte la referencia de [DescribeAvailabilityZones AWS Tools for PowerShellcmdlets](#).

## Python

### SDK para Python (Boto3)

#### Note

Hay más información al respecto. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

```
class AutoScaler:
    """
    Encapsulates Amazon EC2 Auto Scaling and EC2 management actions.
    """

    def __init__(
        self,
        resource_prefix,
        inst_type,
        ami_param,
        autoscaling_client,
        ec2_client,
        ssm_client,
```

```

        iam_client,
    ):
        """
        :param resource_prefix: The prefix for naming AWS resources that are
        created by this class.
        :param inst_type: The type of EC2 instance to create, such as t3.micro.
        :param ami_param: The Systems Manager parameter used to look up the AMI
        that is
                created.
        :param autoscaling_client: A Boto3 EC2 Auto Scaling client.
        :param ec2_client: A Boto3 EC2 client.
        :param ssm_client: A Boto3 Systems Manager client.
        :param iam_client: A Boto3 IAM client.
        """
        self.inst_type = inst_type
        self.ami_param = ami_param
        self.autoscaling_client = autoscaling_client
        self.ec2_client = ec2_client
        self.ssm_client = ssm_client
        self.iam_client = iam_client
        self.launch_template_name = f"{resource_prefix}-template"
        self.group_name = f"{resource_prefix}-group"
        self.instance_policy_name = f"{resource_prefix}-pol"
        self.instance_role_name = f"{resource_prefix}-role"
        self.instance_profile_name = f"{resource_prefix}-prof"
        self.bad_creds_policy_name = f"{resource_prefix}-bc-pol"
        self.bad_creds_role_name = f"{resource_prefix}-bc-role"
        self.bad_creds_profile_name = f"{resource_prefix}-bc-prof"
        self.key_pair_name = f"{resource_prefix}-key-pair"

    def get_availability_zones(self):
        """
        Gets a list of Availability Zones in the AWS Region of the Amazon EC2
        client.

        :return: The list of Availability Zones for the client Region.
        """
        try:
            response = self.ec2_client.describe_availability_zones()
            zones = [zone["ZoneName"] for zone in response["AvailabilityZones"]]
        except ClientError as err:
            raise AutoScalerError(f"Couldn't get availability zones: {err}.")
        else:

```

```
return zones
```

- Para obtener más información sobre la API, consulta [DescribeAvailabilityZones](#) la AWS Referencia de API de SDK for Python (Boto3).

## SAP ABAP

### SDK de SAP ABAP

#### Note

Hay más información al respecto. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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 obtener más información sobre la API, consulte [DescribeAvailabilityZones](#) la referencia sobre la API ABAP del AWS SDK para SAP.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte [Cree recursos de Amazon EC2 mediante un SDK AWS](#). En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **DescribeBundleTasks** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `DescribeBundleTasks`.

### CLI

#### AWS CLI

Para describir las tareas del paquete

En este ejemplo se describen todas las tareas del paquete.

Comando:

```
aws ec2 describe-bundle-tasks
```

Salida:

```
{
  "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 obtener más información sobre la API, consulte [DescribeBundleTasks](#) la Referencia de AWS CLI comandos.



## PowerShell

### Herramientas para PowerShell

Ejemplo 1: en este ejemplo se describe la tarea de agrupamiento especificada.

```
Get-EC2BundleTask -BundleId bun-12345678
```

Ejemplo 2: en este ejemplo se describen las tareas del paquete cuyo estado es «completado» o «fallido».

```
$filter = New-Object Amazon.EC2.Model.Filter
$filter.Name = "state"
$filter.Values = @( "complete", "failed" )

Get-EC2BundleTask -Filter $filter
```

- Para obtener más información sobre la API, consulte la referencia de [DescribeBundleTasks](#) cmdlets AWS Tools for PowerShell .

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **DescribeCapacityReservations** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `DescribeCapacityReservations`.

### CLI

#### AWS CLI

Ejemplo 1: Para describir una o más de sus reservas de capacidad

En el siguiente `describe-capacity-reservations` ejemplo, se muestran detalles sobre todas sus reservas de capacidad en la AWS región actual.

```
aws ec2 describe-capacity-reservations
```

**Salida:**

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

**Ejemplo 2:** Para describir una o más de sus reservas de capacidad

En el siguiente `describe-capacity-reservations` ejemplo, se muestran detalles sobre la reserva de capacidad especificada.

```
aws ec2 describe-capacity-reservations \  
  --capacity-reservation-ids cr-1234abcd56EXAMPLE
```

Salida:

```
{  
  "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 obtener más información, consulte [Visualización de una reserva de capacidad](#) en la Guía del usuario de Amazon Elastic Compute Cloud para instancias de Linux.

- Para obtener más información sobre la API, consulte [DescribeCapacityReservations](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: En este ejemplo se describen una o más de sus reservas de capacidad para la región

```
Get-EC2CapacityReservation -Region eu-west-1
```

**Salida:**

```
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 obtener más información sobre la API, consulte [DescribeCapacityReservations](#) la referencia de AWS Tools for PowerShell cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **DescribeCustomerGateways** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `DescribeCustomerGateways`.

### CLI

#### AWS CLI

Para describir las pasarelas de sus clientes

En este ejemplo, se describen las pasarelas de sus clientes.

Comando:

```
aws ec2 describe-customer-gateways
```

**Salida:**

```
{
  "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 describir una pasarela de clientes específica

En este ejemplo se describe la pasarela de clientes especificada.

Comando:

```
aws ec2 describe-customer-gateways --customer-gateway-ids cgw-0e11f167
```

Salida:

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

- Para obtener más información sobre la API, consulte [DescribeCustomerGateways](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: En este ejemplo se describe la pasarela de clientes especificada.

```
Get-EC2CustomerGateway -CustomerGatewayId cgw-1a2b3c4d
```

Salida:

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

Ejemplo 2: En este ejemplo se describe cualquier pasarela de clientes cuyo estado esté pendiente o disponible.

```
$filter = New-Object Amazon.EC2.Model.Filter
$filter.Name = "state"
$filter.Values = @( "pending", "available" )

Get-EC2CustomerGateway -Filter $filter
```

Ejemplo 3: En este ejemplo se describen todas las pasarelas de clientes.

```
Get-EC2CustomerGateway
```

- Para obtener información sobre la API, consulte la referencia [DescribeCustomerGateways](#) de AWS Tools for PowerShell cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **DescribeDhcpOptions** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `DescribeDhcpOptions`.

### CLI

#### AWS CLI

Ejemplo 1: Para describir las opciones de DHCP

En el siguiente `describe-dhcp-options` ejemplo, se recuperan detalles sobre las opciones de DHCP.

```
aws ec2 describe-dhcp-options
```

Salida:

```
{
  "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 obtener más información, consulte [Trabajar con conjuntos de opciones de DHCP](#) en la Guía del usuario de la AWS VPC.

Ejemplo 2: Para describir las opciones de DHCP y filtrar la salida

En el siguiente `describe-dhcp-options` ejemplo, se describen las opciones de DHCP y se utiliza un filtro para mostrar únicamente las opciones de DHCP disponibles `example.com` para el servidor de nombres de dominio. En el ejemplo, se usa el `--query` parámetro para mostrar solo la información de configuración y el ID en la salida.

```

aws ec2 describe-dhcp-options \
  --filters Name=key,Values=domain-name-servers Name=value,Values=example.com \
  --query "DhcpOptions[*].[DhcpConfigurations,DhcpOptionsId]"

```

Salida:

```

[
  [
    [

```



```

    {
      "Key": "domain-name",
      "Values": [
        {
          "Value": "example.com"
        }
      ]
    },
    {
      "Key": "domain-name-servers",
      "Values": [
        {
          "Value": "172.16.16.16"
        }
      ]
    }
  ],
  "dopt-001122334455667ab"
]

```

Para obtener más información, consulte [Trabajar con conjuntos de opciones de DHCP](#) en la Guía del usuario de la AWS VPC.

- Para obtener más información sobre la API, consulte la Referencia [DescribeDhcpOptions](#) de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: En este ejemplo se enumeran los conjuntos de opciones de DHCP.

```
Get-EC2DhcpOption
```

Salida:

DhcpConfigurations	DhcpOptionsId	Tag
{domain-name, domain-name-servers}	dopt-1a2b3c4d	{}
{domain-name, domain-name-servers}	dopt-2a3b4c5d	{}
{domain-name-servers}	dopt-3a4b5c6d	{}

Ejemplo 2: En este ejemplo se obtienen los detalles de configuración del conjunto de opciones de DHCP especificado.

```
(Get-EC2DhcpOption -DhcpOptionsId dopt-1a2b3c4d).DhcpConfigurations
```

Salida:

Key	Values
---	-----
domain-name	{abc.local}
domain-name-servers	{10.0.0.101, 10.0.0.102}

- Para obtener más información sobre la API, consulte la referencia [DescribeDhcpOptions](#) de AWS Tools for PowerShell cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **DescribeFlowLogs** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar DescribeFlowLogs.

### CLI

#### AWS CLI

Ejemplo 1: Para describir todos los registros de flujo

En el siguiente describe-flow-logs ejemplo, se muestran los detalles de todos los registros de flujo.

```
aws ec2 describe-flow-logs
```

Salida:

```
{
  "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}"
  }
]
}

```

Ejemplo 2: Para describir un subconjunto de sus registros de flujo

En el siguiente `describe-flow-logs` ejemplo, se utiliza un filtro para mostrar los detalles únicamente de los registros de flujo que se encuentran en el grupo de CloudWatch registros especificado en Amazon Logs.

```
aws ec2 describe-flow-logs \
  --filter "Name=log-group-name,Values=MyFlowLogs"
```

- Para obtener más información sobre la API, consulte [DescribeFlowLogs](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: En este ejemplo se describen uno o más registros de flujo con el tipo de destino de registro 's3'

```
Get-EC2FlowLog -Filter @{Name="log-destination-type";Values="s3"}
```

Salida:

```
CreationTime           : 2/25/2019 9:07:36 PM
DeliverLogsErrorMessage :
DeliverLogsPermissionArn :
DeliverLogsStatus      : SUCCESS
FlowLogId              : fl-01b2e3d45f67f8901
FlowLogStatus          : ACTIVE
LogDestination         : arn:aws:s3:::my-bucket-dd-tata
LogDestinationType     : s3
LogGroupName           :
ResourceId              : eni-01d2dda3456b7e890
TrafficType            : ALL
```

- Para obtener más información sobre la API, consulte [DescribeFlowLogs](#) la referencia de AWS Tools for PowerShell cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **DescribeHostReservationOfferings** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `DescribeHostReservationOfferings`.

CLI

### AWS CLI

Para describir las ofertas de reservas dedicadas para anfitriones

En este ejemplo, se describen las reservas de hosts dedicados para la familia de instancias M4 que están disponibles para su compra.

Comando:

```
aws ec2 describe-host-reservation-offerings --filter Name=instance-family,Values=m4
```

Salida:

```
{
  "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 obtener más información sobre la API, consulte [DescribeHostReservationOfferings](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: En este ejemplo se describen las reservas de hosts dedicados que se pueden adquirir para el filtro «instance-family» determinado, donde está PaymentOption «» NoUpfront

```

Get-EC2HostReservationOffering -Filter @{Name="instance-family";Values="m4"} |
Where-Object PaymentOption -eq NoUpfront

```

Salida:

```

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 obtener más información sobre la API, consulte la referencia de [DescribeHostReservationOfferings](#) cmdlets AWS Tools for PowerShell .

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **DescribeHosts** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar DescribeHosts.

### CLI

#### AWS CLI

Para ver los detalles sobre los hosts dedicados

En el siguiente describe-hosts ejemplo, se muestran los detalles de los hosts available dedicados de su AWS cuenta.

```
aws ec2 describe-hosts --filter "Name=state,Values=available"
```

Salida:

```
{  
  "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 obtener más información, consulte [Visualización de hosts dedicados](#) en la Guía del usuario de Amazon Elastic Compute Cloud para instancias de Linux.

- Para obtener más información sobre la API, consulte [DescribeHosts](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: Este ejemplo devuelve los detalles del host EC2

```
Get-EC2Host
```

Salida:



```

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                : {}

```

Ejemplo 2: Este ejemplo consulta el host AvailableInstanceCapacity h-01e23f4cd567899f1

```

Get-EC2Host -HostId h-01e23f4cd567899f1 | Select-Object -ExpandProperty
AvailableCapacity | Select-Object -expand AvailableInstanceCapacity

```

Salida:

```

AvailableCapacity InstanceType TotalCapacity
-----
11                m4.xlarge    11

```

- Para obtener información [DescribeHosts](#) sobre AWS Tools for PowerShell la API, consulte la referencia del cmdlet.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **DescribeIamInstanceProfileAssociations** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `DescribeIamInstanceProfileAssociations`.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en contexto en el siguiente ejemplo de código:

- [Creación y administración de un servicio resiliente](#)

## .NET

### AWS SDK for .NET

#### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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)
{
    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];
}
```

- Para obtener más información sobre la API, consulta [DescribeIamInstanceProfileAssociations](#) la Referencia AWS SDK for .NET de la API.

## CLI

### AWS CLI

Describir las asociaciones de perfiles de instancias de IAM

En este ejemplo, se describen todas las asociaciones de perfiles de instancias de IAM.

Comando:

```
aws ec2 describe-iam-instance-profile-associations
```

Salida:

```
{
  "IamInstanceProfileAssociations": [
    {
      "InstanceId": "i-09eb09efa73ec1dee",
      "State": "associated",
      "AssociationId": "iip-assoc-0db249b1f25fa24b8",
      "IamInstanceProfile": {
        "Id": "AIPAJVQN4F5WVLGCJDRGM",
        "Arn": "arn:aws:iam::123456789012:instance-profile/admin-role"
      }
    },
    {
      "InstanceId": "i-0402909a2f4dff14",
      "State": "associating",
      "AssociationId": "iip-assoc-0d1ec06278d29f44a",
      "IamInstanceProfile": {
        "Id": "AGJAJVQN4F5WVLGCJABCM",
        "Arn": "arn:aws:iam::123456789012:instance-profile/user1-role"
      }
    }
  ]
}
```

- Para obtener más información sobre la API, consulta [DescribeIamInstanceProfileAssociations](#) la Referencia de AWS CLI comandos.

## JavaScript

### SDK para JavaScript (v3)

#### Note

Hay más información. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

```
const ec2Client = new EC2Client({});
const { IamInstanceProfileAssociations } = await ec2Client.send(
  new DescribeIamInstanceProfileAssociationsCommand({
    Filters: [
      { Name: "instance-id", Values: [state.targetInstance.InstanceId] },
    ],
  }),
);
```

- Para obtener más información sobre la API, consulta [DescribeIamInstanceProfileAssociations](#) la Referencia AWS SDK for JavaScript de la API.

## Python

### SDK para Python (Boto3)

#### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

```
class AutoScaler:
    """
    Encapsulates Amazon EC2 Auto Scaling and EC2 management actions.
    """

    def __init__(
        self,
```

```

    resource_prefix,
    inst_type,
    ami_param,
    autoscaling_client,
    ec2_client,
    ssm_client,
    iam_client,
):
    """
    :param resource_prefix: The prefix for naming AWS resources that are
    created by this class.
    :param inst_type: The type of EC2 instance to create, such as t3.micro.
    :param ami_param: The Systems Manager parameter used to look up the AMI
    that is
        created.
    :param autoscaling_client: A Boto3 EC2 Auto Scaling client.
    :param ec2_client: A Boto3 EC2 client.
    :param ssm_client: A Boto3 Systems Manager client.
    :param iam_client: A Boto3 IAM client.
    """
    self.inst_type = inst_type
    self.ami_param = ami_param
    self.autoscaling_client = autoscaling_client
    self.ec2_client = ec2_client
    self.ssm_client = ssm_client
    self.iam_client = iam_client
    self.launch_template_name = f"{resource_prefix}-template"
    self.group_name = f"{resource_prefix}-group"
    self.instance_policy_name = f"{resource_prefix}-pol"
    self.instance_role_name = f"{resource_prefix}-role"
    self.instance_profile_name = f"{resource_prefix}-prof"
    self.bad_creds_policy_name = f"{resource_prefix}-bc-pol"
    self.bad_creds_role_name = f"{resource_prefix}-bc-role"
    self.bad_creds_profile_name = f"{resource_prefix}-bc-prof"
    self.key_pair_name = f"{resource_prefix}-key-pair"

def get_instance_profile(self, instance_id):
    """
    Gets data about the profile associated with an instance.

    :param instance_id: The ID of the instance to look up.
    :return: The profile data.
    """

```

```
    try:
        response =
self.ec2_client.describe_iam_instance_profile_associations(
            Filters=[{"Name": "instance-id", "Values": [instance_id]}]
        )
    except ClientError as err:
        raise AutoScalerError(
            f"Couldn't get instance profile association for instance
{instance_id}: {err}"
        )
    else:
        return response["IamInstanceProfileAssociations"][0]
```

- Para obtener más información sobre la API, consulta [DescribelamInstanceProfileAssociations](#) la AWS Referencia de API de SDK for Python (Boto3).

Para obtener una lista completa de las guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **DescribeIdFormat** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar DescribeIdFormat.

### CLI

#### AWS CLI

Ejemplo 1: Para describir el formato de ID de un recurso

El siguiente describe-id-format ejemplo describe el formato de ID de los grupos de seguridad.

```
aws ec2 describe-id-format \
    --resource security-group
```

En el siguiente resultado de ejemplo, el `Deadline` valor indica que la fecha límite para que este tipo de recurso cambiara permanentemente del formato de ID corto al formato de ID largo expiró a las 00:00 UTC del 15 de agosto de 2018.

```
{
  "Statuses": [
    {
      "Deadline": "2018-08-15T00:00:00.000Z",
      "Resource": "security-group",
      "UseLongIds": true
    }
  ]
}
```

Ejemplo 2: Para describir el formato de ID de todos los recursos

El siguiente `describe-id-format` ejemplo describe el formato de ID de todos los tipos de recursos. Todos los tipos de recursos que admitían el formato de ID corto se cambiaron para usar el formato de ID largo.

```
aws ec2 describe-id-format
```

- Para obtener más información sobre la API, consulte [DescribeIdFormat](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: en este ejemplo se describe el formato de ID del tipo de recurso especificado.

```
Get-EC2IdFormat -Resource instance
```

Salida:

Resource	UseLongIds
-----	-----
instance	False

Ejemplo 2: En este ejemplo se describen los formatos de ID de todos los tipos de recursos que admiten ID más largos.

```
Get-EC2IdFormat
```

Salida:

Resource	UseLongIds
-----	-----
reservation	False
instance	False

- Para obtener más información sobre la API, consulte [DescribeIdFormat](#) la referencia de AWS Tools for PowerShell cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **DescribeIdentityIdFormat** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `DescribeIdentityIdFormat`.

CLI

AWS CLI

Para describir el formato de ID de un rol de IAM

En el siguiente `describe-identity-id-format` ejemplo, se describe el formato de ID que reciben las instancias creadas por el rol de IAM `EC2Role` en su AWS cuenta.

```
aws ec2 describe-identity-id-format \
  --principal-arn arn:aws:iam::123456789012:role/my-iam-role \
  --resource instance
```

El siguiente resultado indica que las instancias creadas por este rol reciben los ID en formato de ID largo.

```
{
```



```

    "Statuses": [
      {
        "Deadline": "2016-12-15T00:00:00Z",
        "Resource": "instance",
        "UseLongIds": true
      }
    ]
  }

```

Para describir el formato de ID de un usuario de IAM

El siguiente `describe-identity-id-format` ejemplo describe el formato de ID que reciben las instantáneas creadas por el usuario de IAM `AdminUser` en su cuenta. AWS

```

aws ec2 describe-identity-id-format \
  --principal-arn arn:aws:iam::123456789012:user/AdminUser \
  --resource snapshot

```

El resultado indica que las instantáneas creadas por este usuario reciben los ID en formato de ID largo.

```

{
  "Statuses": [
    {
      "Deadline": "2016-12-15T00:00:00Z",
      "Resource": "snapshot",
      "UseLongIds": true
    }
  ]
}

```

- Para obtener más información sobre la API, consulte [DescribeIdentityIdFormat](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: Este ejemplo devuelve el formato de ID del recurso «imagen» para el rol asignado

```
Get-EC2IdentityIdFormat -PrincipalArn arn:aws:iam::123456789511:role/JDBC -
Resource image
```

Salida:

```
Deadline                Resource UseLongIds
-----                -
8/2/2018 11:30:00 PM image      True
```

- Para obtener más información sobre la API, consulte la referencia del [DescribeIdentityIdFormat AWS Tools for PowerShell](#) cmdlet.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **DescribeImageAttribute** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar DescribeImageAttribute.

CLI

AWS CLI

Para describir los permisos de lanzamiento de una AMI

En este ejemplo, se describen los permisos de lanzamiento de la AMI especificada.

Comando:

```
aws ec2 describe-image-attribute --image-id ami-5731123e --attribute
launchPermission
```

Salida:

```
{
  "LaunchPermissions": [
    {
      "UserId": "123456789012"
```

```
    }  
  ],  
  "ImageId": "ami-5731123e",  
}
```

Para describir los códigos de producto de una AMI

En este ejemplo se describen los códigos de producto de la AMI especificada. Tenga en cuenta que esta AMI no tiene códigos de producto.

Comando:

```
aws ec2 describe-image-attribute --image-id ami-5731123e --attribute productCodes
```

Salida:

```
{  
  "ProductCodes": [],  
  "ImageId": "ami-5731123e",  
}
```

- Para obtener más información sobre la API, consulte [DescribeImageAttribute](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: en este ejemplo se obtiene la descripción de la AMI especificada.

```
Get-EC2ImageAttribute -ImageId ami-12345678 -Attribute description
```

Salida:

```
BlockDeviceMappings : {}  
Description          : My image description  
ImageId              : ami-12345678  
KernelId             :  
LaunchPermissions    : {}  
ProductCodes         : {}
```

```
RamdiskId      :  
SriovNetSupport :
```

Ejemplo 2: en este ejemplo se obtienen los permisos de lanzamiento de la AMI especificada.

```
Get-EC2ImageAttribute -ImageId ami-12345678 -Attribute launchPermission
```

Salida:

```
BlockDeviceMappings : {}  
Description          :  
ImageId              : ami-12345678  
KernelId             :  
LaunchPermissions    : {all}  
ProductCodes         : {}  
RamdiskId            :  
SriovNetSupport      :
```

Ejemplo 3: En este ejemplo se comprueba si la red mejorada está habilitada.

```
Get-EC2ImageAttribute -ImageId ami-12345678 -Attribute sriovNetSupport
```

Salida:

```
BlockDeviceMappings : {}  
Description          :  
ImageId              : ami-12345678  
KernelId             :  
LaunchPermissions    : {}  
ProductCodes         : {}  
RamdiskId            :  
SriovNetSupport      : simple
```

- Para obtener más información sobre la API, consulte [DescribeImageAttribute](#) la referencia de AWS Tools for PowerShell cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **DescribeImages** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar DescribeImages.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en contexto en el siguiente ejemplo de código:

- [Comience a utilizar instancias](#)

### Bash

#### AWS CLI con el script Bash

#### Note

Hay más información. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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)."
```

Las funciones de utilidad utilizadas en este ejemplo.

```
#####
# 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 obtener más información sobre la API, consulta [DescribeImages](#) la Referencia de AWS CLI comandos.

## CLI

### AWS CLI

#### Ejemplo 1: Describir una AMI

En el siguiente ejemplo de `describe-images`, se describe una AMI especificada en la región especificada.

```
aws ec2 describe-images \
  --region us-east-1 \
  --image-ids ami-1234567890EXAMPLE
```

#### Salida:

```
{
  "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 obtener más información, consulte [Imágenes de máquina de Amazon \(AMI\)](#) en la Guía del usuario de Amazon EC2.

### Ejemplo 2: Describir las AMI con base en filtros

En el siguiente ejemplo de `describe-images`, se describen las AMI de Windows proporcionadas por Amazon y respaldadas por Amazon EBS.

```
aws ec2 describe-images \
  --owners amazon \
  --filters "Name=platform,Values=windows" "Name=root-device-type,Values=ebs"
```

Para ver un ejemplo del resultado de `describe-images`, consulte el ejemplo 1.

Para ver ejemplos adicionales de uso de filtros, consulte [Enumerar y filtrar los recursos](#) en la Guía del usuario de Amazon EC2.

### Ejemplo 3: Describir las AMI con base en etiquetas

En el siguiente ejemplo de `describe-images`, se describen todas las AMI que tienen la etiqueta `Type=Custom`. En el ejemplo se usa el parámetro `--query` para mostrar solo los ID de las AMI.

```
aws ec2 describe-images \  
  --filters "Name=tag:Type,Values=Custom" \  
  --query 'Images[*].[ImageId]' \  
  --output text
```

Salida:

```
ami-1234567890EXAMPLE  
ami-0abcdef1234567890
```

Para ver ejemplos adicionales de uso de filtros de etiquetas, consulte [Trabajo con etiquetas](#) en la Guía del usuario de Amazon EC2.

- Para obtener más información sobre la API, consulte [DescribeImages](#) la Referencia de AWS CLI comandos.

## JavaScript

### SDK para JavaScript (v3)

#### Note

Hay más información. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

```
import { paginateDescribeImages } from "@aws-sdk/client-ec2";  
  
import { client } from "../libs/client.js";  
  
// List at least the first i386 image available for EC2 instances.  
export const main = async () => {  
  // The paginate function is a wrapper around the base command.  
  const paginator = paginateDescribeImages(  
    // Without limiting the page size, this call can take a long time. pageSize  
    // is just sugar for  
    // the MaxResults property in the base command.  
    { client, pageSize: 25 },  
    {  
      // There are almost 70,000 images available. Be specific with your  
      filtering
```

```
// to increase efficiency.
// See https://docs.aws.amazon.com/AWSJavaScriptSDK/v3/latest/clients/
client-ec2/interfaces/describeimagescommandinput.html#filters
Filters: [{ Name: "architecture", Values: ["x86_64"] }],
},
);

try {
  const arm64Images = [];
  for await (const page of paginator) {
    if (page.Images.length) {
      arm64Images.push(...page.Images);
      // Once we have at least 1 result, we can stop.
      if (arm64Images.length >= 1) {
        break;
      }
    }
  }
  console.log(arm64Images);
} catch (err) {
  console.error(err);
}
};
```

- Para obtener más información sobre la API, consulta [DescribeImages](#) la Referencia AWS SDK for JavaScript de la API.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: En este ejemplo se describe la AMI especificada.

```
Get-EC2Image -ImageId ami-12345678
```

Salida:

```
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
```

Ejemplo 2: En este ejemplo se describen las AMI de las que dispone.

```
Get-EC2Image -owner self
```

Ejemplo 3: En este ejemplo se describen las AMI públicas que ejecutan Microsoft Windows Server.

```
Get-EC2Image -Filter @{ Name="platform"; Values="windows" }
```

Ejemplo 4: En este ejemplo se describen todas las AMI públicas de la región «us-west-2».

```
Get-EC2Image -Region us-west-2
```

- Para obtener más información sobre la API, consulte la referencia de [DescribeImages AWS Tools for PowerShell](#) cmdlets.

## Python

### SDK para Python (Boto3)

#### Note

Hay más información al respecto. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

```
class InstanceWrapper:
    """Encapsulates Amazon Elastic Compute Cloud (Amazon EC2) instance
    actions."""

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

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

    def get_images(self, image_ids):
        """
        Gets information about Amazon Machine Images (AMIs) from a list of AMI
        IDs.

        :param image_ids: The list of AMIs to look up.
        :return: A list of Boto3 Image objects that represent the requested AMIs.
        """
        try:
```

```

        images = list(self.ec2_resource.images.filter(ImageIds=image_ids))
    except ClientError as err:
        logger.error(
            "Couldn't get images. Here's why: %s: %s",
            err.response["Error"]["Code"],
            err.response["Error"]["Message"],
        )
        raise
    else:
        return images

```

- Para obtener más información sobre la API, consulta [DescribeImages](#) la AWS Referencia de API de SDK for Python (Boto3).

## Rust

### SDK para Rust

#### Note

Hay más información al respecto. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

```

// A simple Rust program to list all Amazon images in the currently configured
// region.
#[tokio::main]
async fn main() -> Result<(), Error> {
    let config = aws_config::load_from_env().await;
    let client = Client::new(&config);

    let resp = client.describe_images().owners("amazon").send().await?;

    println!("AWS SDK for Rust v{}", PKG_VERSION);
    println!("Describing Amazon Machine Images (AMIs):");

    let mut images: Vec<_> = resp
        .images()
        .iter()

```

```
        .filter(|i| {
            i.description()
                .filter(|i| i.contains("Amazon Linux AMI 2023"))
                .is_some()
        })
        .collect();
images.sort_by(|a, b| a.description.cmp(&b.description));

if images.is_empty() {
    println!("No images found.");
    return Ok(());
}

for image in images {
    let id = image.image_id().unwrap_or_default();
    let description = image.description().unwrap_or_default();

    println!("{id}: {description}");
}

Ok(())
}
```

- Para obtener más información sobre la API, consulta [DescribeImages](#) la referencia sobre la API de AWS SDK para Rust.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte [Cree recursos de Amazon EC2 mediante un SDK AWS](#). En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **DescribeImportImageTasks** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `DescribeImportImageTasks`.

CLI

AWS CLI

Para supervisar una tarea de importación de imágenes

El siguiente describe-import-image-tasks ejemplo comprueba el estado de la tarea de importación de imágenes especificada.

```
aws ec2 describe-import-image-tasks \  
  --import-task-ids import-ami-1234567890abcdef0
```

Resultado de una tarea de importación de imágenes en curso.

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

Resultado de una tarea de importación de imágenes que se ha completado. El ID de la AMI resultante lo proporciona ImageId.

```
{  
  "ImportImageTasks": [  
    {  
      "ImportTaskId": "import-ami-1234567890abcdef0",  
      "ImageId": "ami-1234567890abcdef0",  
      "SnapshotDetails": [  
        {  
          "DiskImageSize": 705638400.0,  
          "Format": "ova",  
          "Status": "completed",  
          "UserBucket": {  
            "S3Bucket": "my-import-bucket",  
            "S3Key": "vms/my-server-vm.ova"  
          }  
        }  
      ],  
      "Status": "completed",  
      "StatusMessage": ""  
    }  
  ]  
}
```



```

        "SnapshotId": "snap-1234567890abcdef0"
        "Status": "completed",
        "UserBucket": {
            "S3Bucket": "my-import-bucket",
            "S3Key": "vms/my-server-vm.ova"
        }
    },
    "Status": "completed"
}
]
}

```

- Para obtener más información sobre la API, consulte [DescribeImportImageTasks](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: En este ejemplo se describe la tarea de importación de imágenes especificada.

```
Get-EC2ImportImageTask -ImportTaskId import-ami-hgfedcba
```

Salida:

```

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     :

```

Ejemplo 2: En este ejemplo se describen todas las tareas de importación de imágenes.

```
Get-EC2ImportImageTask
```

**Salida:**

```
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 obtener más información sobre la API, consulte [DescribeImportImageTasks](#) la referencia de AWS Tools for PowerShell cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **DescribeImportSnapshotTasks** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `DescribeImportSnapshotTasks`.

### CLI

#### AWS CLI

Para supervisar una tarea de importación de instantáneas

El siguiente `describe-import-snapshot-tasks` ejemplo comprueba el estado de la tarea de importación de instantáneas especificada.

```
aws ec2 describe-import-snapshot-tasks \  
  --import-task-ids import-snap-1234567890abcdef0
```

Resultado de una tarea de importación de instantáneas en curso:

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

Resultado de una tarea de importación de instantáneas que se ha completado. El ID de la instantánea resultante lo proporciona `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 obtener más información sobre la API, consulte [DescribeImportSnapshotTasks](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: en este ejemplo se describe la tarea de importación de instantáneas especificada.

```
Get-EC2ImportSnapshotTask -ImportTaskId import-snap-abcdefgh
```

Salida:

Description	ImportTaskId	SnapshotTaskDetail
-----	-----	-----
Disk Image Import 1	import-snap-abcdefgh	Amazon.EC2.Model.SnapshotTaskDetail

Ejemplo 2: En este ejemplo se describen todas las tareas de importación de instantáneas.

```
Get-EC2ImportSnapshotTask
```

Salida:

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 obtener más información sobre la API, consulte [DescribeImportSnapshotTasks](#) la referencia de AWS Tools for PowerShell cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **DescribeInstanceAttribute** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `DescribeInstanceAttribute`.

### CLI

#### AWS CLI

Para describir el tipo de instancia

En este ejemplo, se describe el tipo de instancia de la instancia especificada.

Comando:

```
aws ec2 describe-instance-attribute --instance-id i-1234567890abcdef0 --attribute  
instanceType
```

Salida:

```
{  
  "InstanceId": "i-1234567890abcdef0"  
  "InstanceType": {  
    "Value": "t1.micro"  
  }  
}
```

Para describir el `disableApiTermination` atributo

En este ejemplo se describe el `disableApiTermination` atributo de la instancia especificada.

**Comando:**

```
aws ec2 describe-instance-attribute --instance-id i-1234567890abcdef0 --attribute
disableApiTermination
```

**Salida:**

```
{
  "InstanceId": "i-1234567890abcdef0"
  "DisableApiTermination": {
    "Value": "false"
  }
}
```

Para describir el mapeo de dispositivos de bloques de una instancia

En este ejemplo, se describe el `blockDeviceMapping` atributo de la instancia especificada.

**Comando:**

```
aws ec2 describe-instance-attribute --instance-id i-1234567890abcdef0 --attribute
blockDeviceMapping
```

**Salida:**

```
{
  "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 obtener más información sobre la API, consulte [DescribeInstanceAttribute](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: en este ejemplo se describe el tipo de instancia de la instancia especificada.

```
Get-EC2InstanceAttribute -InstanceId i-12345678 -Attribute instanceType
```

Salida:

```
InstanceType           : t2.micro
```

Ejemplo 2: En este ejemplo se describe si la red mejorada está habilitada para la instancia especificada.

```
Get-EC2InstanceAttribute -InstanceId i-12345678 -Attribute sriovNetSupport
```

Salida:

```
SriovNetSupport       : simple
```

Ejemplo 3: en este ejemplo se describen los grupos de seguridad de la instancia especificada.

```
(Get-EC2InstanceAttribute -InstanceId i-12345678 -Attribute groupSet).Groups
```

Salida:

```
GroupId
-----
```

```
sg-12345678  
sg-45678901
```

Ejemplo 4: En este ejemplo se describe si la optimización de EBS está habilitada para la instancia especificada.

```
Get-EC2InstanceAttribute -InstanceId i-12345678 -Attribute ebsOptimized
```

Salida:

```
EbsOptimized : False
```

Ejemplo 5: en este ejemplo se describe el atributo `disableApiTermination` de la instancia especificada.

```
Get-EC2InstanceAttribute -InstanceId i-12345678 -Attribute disableApiTermination
```

Salida:

```
DisableApiTermination : False
```

Ejemplo 6: En este ejemplo se describe el atributo `instanceInitiatedShutdownBehavior` de la instancia especificada.

```
Get-EC2InstanceAttribute -InstanceId i-12345678 -Attribute  
instanceInitiatedShutdownBehavior
```

Salida:

```
InstanceInitiatedShutdownBehavior : stop
```

- Para obtener más información sobre la API, consulte la referencia de [DescribeInstanceAttribute AWS Tools for PowerShell cmdlets](#).

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.



## Úselo `DescribeInstanceStatus` con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `DescribeInstanceStatus`.

### CLI

#### AWS CLI

Describir el estado de las instancias

En el siguiente ejemplo de `describe-instance-status`, se muestran los detalles de la instancia especificada.

```
aws ec2 describe-instance-status \  
  --instance-ids i-1234567890abcdef0
```

Salida:

```
{  
  "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 obtener más información, consulte [Monitorear el estado de las instancias](#) en la Guía del usuario de Amazon EC2.

- Para obtener más información sobre la API, consulte [DescribeInstanceStatus](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: en este ejemplo se describe el estado de la instancia especificada.

```
Get-EC2InstanceStatus -InstanceId i-12345678
```

Salida:

```

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

```

Salida:

```

Code    Name
----    -
16     running

```

```
$status.Status
```

**Salida:**

```

Details          Status
-----          -
{reachability}  ok

```

```
$status.SystemStatus
```

**Salida:**

```

Details          Status
-----          -
{reachability}  ok

```

- Para obtener información sobre la API, consulte [DescribeInstanceStatus](#) la referencia del AWS Tools for PowerShell cmdlet.

**Rust****SDK para Rust****Note**

Hay más información al respecto. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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 obtener más información sobre la API, consulta [DescribeInstanceStatus](#) la referencia sobre la API de AWS SDK para Rust.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte [Cree recursos de Amazon EC2 mediante un SDK AWS](#). En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **DescribeInstanceTypes** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `DescribeInstanceTypes`.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en contexto en el siguiente ejemplo de código:

- [Comience a utilizar instancias](#)

## .NET

### AWS SDK for .NET

#### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

```
/// <summary>
/// Describe the instance types available.
/// </summary>
/// <returns>A list of instance type information.</returns>
public async Task<List<InstanceTypeInfo>>
DescribeInstanceTypes(ArchitectureValues architecture)
{
    var request = new DescribeInstanceTypesRequest();

    var filters = new List<Filter>
        { new Filter("processor-info.supported-architecture", new
List<string> { architecture.ToString() }) };
    filters.Add(new Filter("instance-type", new() { "*.micro", "*.small" }));

    request.Filters = filters;
    var instanceTypes = new List<InstanceTypeInfo>();

    var paginator = _amazonEC2.Paginators.DescribeInstanceTypes(request);
    await foreach (var instanceType in paginator.InstanceTypes)
    {
        instanceTypes.Add(instanceType);
    }
    return instanceTypes;
}
```

- Para obtener más información sobre la API, consulta [DescribeInstanceTypes](#) la Referencia AWS SDK for .NET de la API.

## Bash

## AWS CLI con el script Bash

 Note

Hay más información. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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
}

```

Las funciones de utilidad utilizadas en este ejemplo.

```

#####
# 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 obtener más información sobre la API, consulta [DescribeInstanceTypes](#) la Referencia de AWS CLI comandos.

## CLI

## AWS CLI

## Ejemplo 1: Describir un tipo de instancia

En el ejemplo siguiente de `describe-instance-types`, se muestran los detalles del tipo de instancia especificado.

```
aws ec2 describe-instance-types \  
  --instance-types t2.micro
```

## Salida:

```
{  
  "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,
        "EfaSupport": "unsupported"
    },
    "PlacementGroupInfo": {
        "SupportedStrategies": [
            "partition",
            "spread"
        ]
    },
    "HibernationSupported": false,
    "BurstablePerformanceSupported": true,
    "DedicatedHostsSupported": false,
    "AutoRecoverySupported": true
    }
}
]
```

Para obtener más información, consulte [Tipos de instancias](#) en la Guía del usuario de Amazon Elastic Compute Cloud para instancias de Linux.

### Ejemplo 2: Filtrar los tipos de instancias disponibles

Puede especificar un filtro para limitar los resultados a los tipos de instancias que tienen una característica específica. En el siguiente ejemplo de `describe-instance-types`, se enumeran los tipos de instancias que admiten la hibernación.

```
aws ec2 describe-instance-types \  
  --filters Name=hibernation-supported,Values=true --query  
  'InstanceTypes[*].InstanceType'
```

Salida:

```
[  
  "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 obtener más información, consulte [Tipos de instancias](#) en la Guía del usuario de Amazon Elastic Compute Cloud para instancias de Linux.

- Para obtener más información sobre la API, consulte [DescribeInstanceTypes](#) la Referencia de AWS CLI comandos.

## Java

### SDK para Java 2.x

#### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

```
// Get a list of instance types.
```

```
public static String getInstanceTypes(Ec2Client ec2) {
    String instanceType;
    try {
        DescribeInstanceTypesRequest typesRequest =
DescribeInstanceTypesRequest.builder()
            .maxResults(10)
            .build();

        DescribeInstanceTypesResponse response =
ec2.describeInstanceTypes(typesRequest);
        List<InstanceTypeInfo> instanceTypes = response.getInstanceTypes();
        for (InstanceTypeInfo type : instanceTypes) {
            System.out.println("The memory information of this type is " +
type.getMemoryInfo().getSizeInMiB());
            System.out.println("Network information is " +
type.getNetworkInfo().toString());
            System.out.println("Instance type is " +
type.getInstanceType().toString());
            instanceType = type.getInstanceType().toString();
            if (instanceType.compareTo("t2.2xlarge") == 0){
                return instanceType;
            }
        }

    } catch (SsmException e) {
        System.err.println(e.getMessage());
        System.exit(1);
    }
    return "";
}
```

- Para obtener más información sobre la API, consulta [DescribeInstanceTypes](#) la Referencia AWS SDK for Java 2.x de la API.

## JavaScript

### SDK para JavaScript (v3)

#### Note

Hay más información. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

```
import {
  paginateDescribeInstanceTypes,
  DescribeInstanceTypesCommand,
} from "@aws-sdk/client-ec2";

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

// List at least the first arm64 EC2 instance type available.
export const main = async () => {
  // The paginate function is a wrapper around the underlying command.
  const paginator = paginateDescribeInstanceTypes(
    // Without limiting the page size, this call can take a long time. pageSize
    // is just sugar for
    // the MaxResults property in the underlying command.
    { client, pageSize: 25 },
    {
      Filters: [
        { Name: "processor-info.supported-architecture", Values: ["x86_64"] },
        { Name: "free-tier-eligible", Values: ["true"] },
      ],
    }
  );

  try {
    const instanceTypes = [];

    for await (const page of paginator) {
      if (page.InstanceTypes.length) {
        instanceTypes.push(...page.InstanceTypes);

        // When we have at least 1 result, we can stop.
        if (instanceTypes.length >= 1) {
```

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

- Para obtener más información sobre la API, consulta [DescribeInstanceTypes](#) la Referencia AWS SDK for JavaScript de la API.

## Kotlin

### SDK para Kotlin

#### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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 obtener más información sobre la API, consulta [DescribeInstanceTypes](#) la referencia sobre el AWS SDK para la API de Kotlin.

## Python

### SDK para Python (Boto3)

#### Note

Hay más información al respecto. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

```

class InstanceWrapper:
    """Encapsulates Amazon Elastic Compute Cloud (Amazon EC2) instance
    actions."""

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

```



```
self.instance = instance

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

def get_instance_types(self, architecture):
    """
    Gets instance types that support the specified architecture and are
    designated
    as either 'micro' or 'small'. When an instance is created, the instance
    type
    you specify must support the architecture of the AMI you use.

    :param architecture: The kind of architecture the instance types must
    support,
                        such as 'x86_64'.
    :return: A list of instance types that support the specified architecture
    and are either 'micro' or 'small'.
    """
    try:
        inst_types = []
        it_paginator = self.ec2_resource.meta.client.get_paginator(
            "describe_instance_types"
        )
        for page in it_paginator.paginate(
            Filters=[
                {
                    "Name": "processor-info.supported-architecture",
                    "Values": [architecture],
                },
                {"Name": "instance-type", "Values": ["*.micro", "*.small"]},
            ]
        ):
            inst_types += page["InstanceTypes"]
    except ClientError as err:
        logger.error(
            "Couldn't get instance types. Here's why: %s: %s",
            err.response["Error"]["Code"],
            err.response["Error"]["Message"],
        )
    raise
```

```
else:
    return inst_types
```

- Para obtener más información sobre la API, consulta [DescribeInstanceTypes](#) la AWS Referencia de API de SDK for Python (Boto3).

Para obtener una lista completa de las guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **DescribeInstances** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar DescribeInstances.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en contexto en los siguientes ejemplos de código:

- [Creación y administración de un servicio resiliente](#)
- [Comience a utilizar instancias](#)

.NET

AWS SDK for .NET

### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

```
/// <summary>
/// Get information about existing EC2 images.
/// </summary>
/// <returns>Async task.</returns>
public async Task DescribeInstances()
{
    // List all EC2 instances.
    await GetInstanceDescriptions();
```

```
        string tagName = "IncludeInList";
        string tagValue = "Yes";
        await GetInstanceDescriptionsFiltered(tagName, tagValue);
    }

    /// <summary>
    /// Get information for all existing Amazon EC2 instances.
    /// </summary>
    /// <returns>Async task.</returns>
    public async Task GetInstanceDescriptions()
    {
        Console.WriteLine("Showing all instances:");
        var paginator = _amazonEC2.Paginators.DescribeInstances(new
DescribeInstancesRequest());

        await foreach (var response in paginator.Responses)
        {
            foreach (var reservation in response.Reservations)
            {
                foreach (var instance in reservation.Instances)
                {
                    Console.Write($"Instance ID: {instance.InstanceId}");
                    Console.WriteLine($"\\tCurrent State: {instance.State.Name}");
                }
            }
        }
    }

    /// <summary>
    /// Get information about EC2 instances filtered by a tag name and value.
    /// </summary>
    /// <param name="tagName">The name of the tag to filter on.</param>
    /// <param name="tagValue">The value of the tag to look for.</param>
    /// <returns>Async task.</returns>
    public async Task GetInstanceDescriptionsFiltered(string tagName, string
tagValue)
    {
        // This tag filters the results of the instance list.
        var filters = new List<Filter>
        {
            new Filter
            {
                Name = $"tag:{tagName}",
```

```
        Values = new List<string>
        {
            tagValue,
        },
    },
};
var request = new DescribeInstancesRequest
{
    Filters = filters,
};

Console.WriteLine("\nShowing instances with tag: \"IncludeInList\" set to
\"Yes\".");
var paginator = _amazonEC2.Paginators.DescribeInstances(request);

await foreach (var response in paginator.Responses)
{
    foreach (var reservation in response.Reservations)
    {
        foreach (var instance in reservation.Instances)
        {
            Console.WriteLine($"Instance ID: {instance.InstanceId} ");
            Console.WriteLine($"Current State: {instance.State.Name}");
        }
    }
}
```

- Para obtener más información sobre la API, consulta [DescribeInstances](#) la Referencia AWS SDK for .NET de la API.

## Bash

### AWS CLI con el script Bash

#### Note

Hay más información. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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
}
```

```

        ;;
    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
}

```

Las funciones de utilidad utilizadas en este ejemplo.

```

#####
# 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 obtener más información sobre la API, consulta [DescribeInstances](#) la Referencia de AWS CLI comandos.

## C++

## SDK para C++

 Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

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

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

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

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

                Aws::String typeString =
```



```

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

    Aws::String monitorString =

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

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

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

```

- Para obtener más información sobre la API, consulta [DescribeInstances](#) la Referencia AWS SDK for C++ de la API.

## CLI

### AWS CLI

#### Ejemplo 1: Describir una instancia

En el siguiente ejemplo de `describe-instances`, se describe la instancia especificada.

```
aws ec2 describe-instances \  
  --instance-ids i-1234567890abcdef0
```

#### Salida:

```
{  
  "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": "11111111111111111111"
    },
    "NetworkInterfaces": [
      {
        "Association": {
          "IpOwnerId": "amazon",
          "PublicDnsName": "ec2-34-253-223-13.us-
east-2.compute.amazonaws.com",
          "PublicIp": "34.253.223.13"
        },
        "Attachment": {
          "AttachTime": "2022-11-15T10:48:59+00:00",
          "AttachmentId": "eni-attach-1234567890abcdefg",
          "DeleteOnTermination": true,
          "DeviceIndex": 0,
```

```

        "Status": "attached",
        "NetworkCardIndex": 0
    },
    "Description": "",
    "Groups": [
        {
            "GroupName": "launch-wizard-146",
            "GroupId": "sg-1234567890abcdefg"
        }
    ],
    "Ipv6Addresses": [],
    "MacAddress": "00:11:22:33:44:55",
    "NetworkInterfaceId": "eni-1234567890abcdefg",
    "OwnerId": "104024344472",
    "PrivateDnsName": "ip-10-0-0-157.us-
east-2.compute.internal",
    "PrivateIpAddress": "10-0-0-157",
    "PrivateIpAddresses": [
        {
            "Association": {
                "IpOwnerId": "amazon",
                "PublicDnsName": "ec2-34-253-223-13.us-
east-2.compute.amazonaws.com",
                "PublicIp": "34.253.223.13"
            },
            "Primary": true,
            "PrivateDnsName": "ip-10-0-0-157.us-
east-2.compute.internal",
            "PrivateIpAddress": "10-0-0-157"
        }
    ],
    "SourceDestCheck": true,
    "Status": "in-use",
    "SubnetId": "subnet-1234567890abcdefg",
    "VpcId": "vpc-1234567890abcdefg",
    "InterfaceType": "interface"
    }
],
"RootDeviceName": "/dev/xvda",
"RootDeviceType": "ebs",
"SecurityGroups": [
    {
        "GroupName": "launch-wizard-146",
        "GroupId": "sg-1234567890abcdefg"
    }
]

```

```
    }
  ],
  "SourceDestCheck": true,
  "Tags": [
    {
      "Key": "Name",
      "Value": "my-instance"
    }
  ],
  "VirtualizationType": "hvm",
  "CpuOptions": {
    "CoreCount": 1,
    "ThreadsPerCore": 2
  },
  "CapacityReservationSpecification": {
    "CapacityReservationPreference": "open"
  },
  "HibernationOptions": {
    "Configured": false
  },
  "MetadataOptions": {
    "State": "applied",
    "HttpTokens": "optional",
    "HttpPutResponseHopLimit": 1,
    "HttpEndpoint": "enabled",
    "HttpProtocolIpv6": "disabled",
    "InstanceMetadataTags": "enabled"
  },
  "EnclaveOptions": {
    "Enabled": false
  },
  "PlatformDetails": "Linux/UNIX",
  "UsageOperation": "RunInstances",
  "UsageOperationUpdateTime": "2022-11-15T10:48:59+00:00",
  "PrivateDnsNameOptions": {
    "HostnameType": "ip-name",
    "EnableResourceNameDnsARecord": true,
    "EnableResourceNameDnsAAAARecord": false
  },
  "MaintenanceOptions": {
    "AutoRecovery": "default"
  }
}
],
```

```
        "OwnerId": "111111111111",
        "ReservationId": "r-1234567890abcdefg"
    }
]
}
```

### Ejemplo 2: Filtrar instancias con el tipo especificado

En el siguiente ejemplo de `describe-instances`, se usan filtros para limitar los resultados a las instancias del tipo especificado.

```
aws ec2 describe-instances \
    --filters Name=instance-type,Values=m5.large
```

Para ver un ejemplo del resultado, consulte el ejemplo 1.

Para obtener más información, consulte [Enumerar y filtrar mediante la CLI](#) en la Guía del usuario de Amazon EC2.

### Ejemplo 3: Filtrar instancias con el tipo y la zona de disponibilidad especificados

En el siguiente ejemplo de `describe-instances`, se usan varios filtros para limitar los resultados a las instancias del tipo especificado que también se encuentran en la zona de disponibilidad especificada.

```
aws ec2 describe-instances \
    --filters Name=instance-type,Values=t2.micro,t3.micro Name=availability-
zone,Values=us-east-2c
```

Para ver un ejemplo del resultado, consulte el ejemplo 1.

### Ejemplo 4: Filtrar instancias con el tipo y la zona de disponibilidad especificados mediante un archivo JSON

En el siguiente ejemplo de `describe-instances`, se usa un archivo de entrada JSON para realizar el mismo filtrado que en el ejemplo anterior. Cuando los filtros se vuelven más complicados, es más fácil especificarlos en un archivo JSON.

```
aws ec2 describe-instances \
    --filters file://filters.json
```

## Contenidos de `filters.json`:

```
[
  {
    "Name": "instance-type",
    "Values": ["t2.micro", "t3.micro"]
  },
  {
    "Name": "availability-zone",
    "Values": ["us-east-2c"]
  }
]
```

Para ver un ejemplo del resultado, consulte el ejemplo 1.

### Ejemplo 5: Filtrar instancias con la etiqueta de propietario especificada

En el siguiente ejemplo de `describe-instances`, se usan filtros de etiquetas para limitar los resultados a las instancias que tienen una etiqueta con la clave de etiqueta especificada (propietario), independientemente del valor de la etiqueta.

```
aws ec2 describe-instances \
  --filters "Name=tag-key,Values=Owner"
```

Para ver un ejemplo del resultado, consulte el ejemplo 1.

### Ejemplo 6: Filtrar las instancias con el valor de etiqueta de mi equipo especificado

En el siguiente ejemplo de `describe-instances`, se usan filtros de etiquetas para limitar los resultados a las instancias que tienen una etiqueta con el valor de etiqueta especificado (mi equipo), independientemente de la clave de la etiqueta.

```
aws ec2 describe-instances \
  --filters "Name=tag-value,Values=my-team"
```

Para ver un ejemplo del resultado, consulte el ejemplo 1.

### Ejemplo 7: Filtrar las instancias con la etiqueta de propietario y el valor de mi equipo especificados

En el siguiente ejemplo de `describe-instances`, se usan filtros de etiquetas para limitar los resultados a las instancias que tienen la etiqueta especificada (propietario = mi equipo).

```
aws ec2 describe-instances \  
  --filters "Name=tag:Owner,Values=my-team"
```

Para ver un ejemplo del resultado, consulte el ejemplo 1.

### Ejemplo 8: Mostrar solo los ID de instancia y subred de todas las instancias

En los siguientes ejemplos de `describe-instances`, se usa el parámetro `--query` para mostrar solo los ID de instancia y subred de todas las instancias, en formato JSON.

Linux y 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
```

Salida:

```
[  
  {  
    "Instance": "i-057750d42936e468a",  
    "Subnet": "subnet-069beee9b12030077"  
  },  
  {  
    "Instance": "i-001efd250faaa6ffa",  
    "Subnet": "subnet-0b715c6b7db68927a"  
  },  
  {  
    "Instance": "i-027552a73f021f3bd",  
    "Subnet": "subnet-0250c25a1f4e15235"  
  }  
  ...  
]
```



]

**Ejemplo 9: Filtrar instancias del tipo especificado y mostrar solo sus ID de instancia**

En el siguiente ejemplo de `describe-instances`, se usan filtros para limitar los resultados a las instancias del tipo especificado y el parámetro `--query` para mostrar solo los ID de las instancias.

```
aws ec2 describe-instances \
  --filters "Name=instance-type,Values=t2.micro" \
  --query "Reservations[*].Instances[*].[InstanceId]" \
  --output text
```

Salida:

```
i-031c0dc19de2fb70c
i-00d8bfff789a736b75
i-0b715c6b7db68927a
i-0626d4edd54f1286d
i-00b8ae04f9f99908e
i-0fc71c25d2374130c
```

**Ejemplo 10: Filtrar instancias del tipo especificado y mostrar solo sus ID de instancia, la zona de disponibilidad y el valor de etiqueta especificado**

En los siguientes ejemplos de `describe-instances`, se muestran el ID de la instancia, la zona de disponibilidad y el valor de la etiqueta `Name` para las instancias que tienen una etiqueta con el nombre `tag-key`, en formato de tabla.

Linux y 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

```

Salida:

```

-----
|                               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   |
+-----+-----+-----+-----+

```

Ejemplo 11: Describir las instancias de un grupo con ubicación en particiones

En el siguiente ejemplo de `describe-instances`, se describe la instancia especificada. El resultado incluye la información de ubicación de la instancia, la cual contiene el nombre del grupo con ubicación y el número de partición de la instancia.

```

aws ec2 describe-instances \
  --instance-ids i-0123a456700123456 \
  --query "Reservations[*].Instances[*].Placement"

```

Salida:

```

[
  [
    {
      "AvailabilityZone": "us-east-1c",
      "GroupName": "HDFS-Group-A",
      "PartitionNumber": 3,
      "Tenancy": "default"
    }
  ]
]

```

Para obtener más información, consulte [Descripción de instancias en un grupo con ubicación](#) en la Guía del usuario de Amazon EC2.

Ejemplo 12: Filtrar las instancias con el grupo con ubicación y el número de partición especificados

En el siguiente ejemplo de `describe-instances`, se filtran los resultados solo para las instancias con el grupo con ubicación y el número de partición especificados.

```
aws ec2 describe-instances \  
  --filters "Name=placement-group-name,Values=HDFS-Group-A" "Name=placement-  
partition-number,Values=7"
```

A continuación, se muestra solo la información relevante de la salida.

```
"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 obtener más información, consulte [Descripción de instancias en un grupo con ubicación](#) en la Guía del usuario de Amazon EC2.

Ejemplo 13: Filtrar las instancias que están configuradas para permitir el acceso a las etiquetas desde los metadatos de la instancia

En el siguiente ejemplo de `describe-instances`, se filtran los resultados solo para las instancias que están configuradas para permitir el acceso a las etiquetas de la instancia desde los metadatos de la instancia.

```
aws ec2 describe-instances \  
  --filters "Name=metadata-options.instance-metadata-tags,Values=enabled" \  
  --query "Reservations[*].Instances[*].InstanceId" \  
  --output text
```

El resultado esperado es el siguiente.

```
i-1234567890abcdefg  
i-abcdefg1234567890  
i-111111111aaaaaaaa  
i-aaaaaaaa111111111
```

Para obtener más información, consulte [Trabajo con etiquetas de instancia en los metadatos de la instancia](#) en la Guía del usuario de Amazon EC2.

- Para obtener más información sobre la API, consulta [DescribeInstances](#) la Referencia de AWS CLI comandos.

## Java

### SDK para Java 2.x

#### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

```
import software.amazon.awssdk.regions.Region;  
import software.amazon.awssdk.services.ec2.Ec2Client;  
import software.amazon.awssdk.services.ec2.model.DescribeInstancesRequest;  
import software.amazon.awssdk.services.ec2.model.Ec2Exception;  
import software.amazon.awssdk.services.ec2.paginators.DescribeInstancesIterable;  
  
/**  
 * Before running this Java V2 code example, set up your development
```

```
* environment, including your credentials.
*
* For more information, see the following documentation topic:
*
* https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
*/
public class DescribeInstances {
    public static void main(String[] args) {
        Region region = Region.US_EAST_1;
        Ec2Client ec2 = Ec2Client.builder()
            .region(region)
            .build();

        describeEC2Instances(ec2);
        ec2.close();
    }

    public static void describeEC2Instances(Ec2Client ec2) {
        try {
            DescribeInstancesRequest request = DescribeInstancesRequest.builder()
                .maxResults(10)
                .build();

            DescribeInstancesIterable instancesIterable =
                ec2.describeInstancesPaginator(request);
            instancesIterable.stream()
                .flatMap(r -> r.reservations().stream())
                .flatMap(reservation -> reservation.instances().stream())
                .forEach(instance -> {
                    System.out.println("Instance Id is " +
instance.instanceId());
                    System.out.println("Image id is " + instance.imageId());
                    System.out.println("Instance type is " +
instance.instanceType());
                    System.out.println("Instance state name is " +
instance.state().name());
                    System.out.println("Monitoring information is " +
instance.monitoring().state());
                });
        } catch (Ec2Exception e) {
            System.err.println(e.awsErrorDetails().errorCode());
            System.exit(1);
        }
    }
}
```

```
    }  
  }  
}
```

- Para obtener más información sobre la API, consulta [DescribeInstances](#) la Referencia AWS SDK for Java 2.x de la API.

## JavaScript

### SDK para JavaScript (v3)

#### Note

Hay más información. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

```
import { DescribeInstancesCommand } from "@aws-sdk/client-ec2";  
  
import { client } from "../libs/client.js";  
  
// List all of your EC2 instances running with x86_64 architecture that were  
// launched this month.  
export const main = async () => {  
  const d = new Date();  
  const year = d.getFullYear();  
  const month = `0${d.getMonth() + 1}`.slice(-2);  
  const launchTimePattern = `${year}-${month}-*`;  
  const command = new DescribeInstancesCommand({  
    Filters: [  
      { Name: "architecture", Values: ["x86_64"] },  
      { Name: "instance-state-name", Values: ["running"] },  
      {  
        Name: "launch-time",  
        Values: [launchTimePattern],  
      },  
    ],  
  });  
  
  try {
```

```

const { Reservations } = await client.send(command);
const instanceList = Reservations.reduce((prev, current) => {
  return prev.concat(current.Instances);
}, []);

console.log(instanceList);
} catch (err) {
  console.error(err);
}
};

```

- Para obtener más información sobre la API, consulta [DescribeInstances](#) la Referencia AWS SDK for JavaScript de la API.

## Kotlin

### SDK para Kotlin

#### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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 obtener más información sobre la API, consulta [DescribeInstances](#) la referencia sobre el AWS SDK para la API de Kotlin.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: en este ejemplo se describe la instancia especificada.

```
(Get-EC2Instance -InstanceId i-12345678).Instances
```

Salida:

```

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

```

Ejemplo 2: en este ejemplo se describen todas las instancias de la región actual, agrupadas por reserva. Para ver los detalles de la instancia, amplíe la colección de instancias dentro de cada objeto de reserva.

```
Get-EC2Instance
```

Salida:

```

GroupNames    : {}
Groups        : {}
Instances     : {}
OwnerId       : 123456789012
RequesterId   : 226008221399
ReservationId : r-c5df370c

GroupNames    : {}
Groups        : {}
Instances     : {}
OwnerId       : 123456789012
RequesterId   : 854251627541
ReservationId : r-63e65bab
...

```

Ejemplo 3: Este ejemplo ilustra el uso de un filtro para consultar instancias de EC2 en una subred específica de una VPC.

```
(Get-EC2Instance -Filter @{Name="vpc-id";Values="vpc-1a2bc34d"},@{Name="subnet-id";Values="subnet-1a2b3c4d"}).Instances
```

Salida:

```
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 obtener información sobre la API, consulte la referencia de [DescribeInstances](#) cmdlets AWS Tools for PowerShell .

## Python

### SDK para Python (Boto3)

#### Note

Hay más información al respecto. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

```
class InstanceWrapper:
    """Encapsulates Amazon Elastic Compute Cloud (Amazon EC2) instance
    actions."""

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

```
    """
    self.ec2_resource = ec2_resource
    self.instance = instance

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

    def display(self, indent=1):
        """
        Displays information about an instance.

        :param indent: The visual indent to apply to the output.
        """
        if self.instance is None:
            logger.info("No instance to display.")
            return

        try:
            self.instance.load()
            ind = "\t" * indent
            print(f"{ind}ID: {self.instance.id}")
            print(f"{ind}Image ID: {self.instance.image_id}")
            print(f"{ind}Instance type: {self.instance.instance_type}")
            print(f"{ind}Key name: {self.instance.key_name}")
            print(f"{ind}VPC ID: {self.instance.vpc_id}")
            print(f"{ind}Public IP: {self.instance.public_ip_address}")
            print(f"{ind}State: {self.instance.state['Name']}")
        except ClientError as err:
            logger.error(
                "Couldn't display your instance. Here's why: %s: %s",
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

- Para obtener más información sobre la API, consulta [DescribeInstances](#) la AWS Referencia de API de SDK for Python (Boto3).

## Ruby

### SDK para Ruby

#### Note

Hay más información al respecto. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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 obtener más información sobre la API, consulta [DescribeInstances](#) la Referencia AWS SDK for Ruby de la API.

## Rust

### SDK para Rust

#### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

```

async fn show_state(client: &Client, ids: Option<Vec<String>>) -> Result<(),
Error> {
    let resp = client
        .describe_instances()
        .set_instance_ids(ids)
        .send()
        .await?;

    for reservation in resp.reservations() {
        for instance in reservation.instances() {
            println!("Instance ID: {}", instance.instance_id().unwrap());
            println!(
                "State:      {:?}",
                instance.state().unwrap().name().unwrap()
            );
            println!();
        }
    }
}

```

```

    }
  }

  Ok(())
}

```

- Para obtener más información sobre la API, consulta [DescribeInstances](#) la referencia sobre la API de AWS SDK para Rust.

## SAP ABAP

### SDK de SAP ABAP

#### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

```

TRY.
  oo_result = lo_ec2->describeinstances( ) .
  oo_result is returned for testing purposes. "

  " Retrieving details of EC2 instances. "
  DATA: lv_instance_id   TYPE /aws1/ec2string,
         lv_status       TYPE /aws1/ec2instancename,
         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_instance_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 obtener más información sobre la API, consulte [DescribeInstances](#) la referencia sobre la API ABAP del AWS SDK para SAP.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte [Cree recursos de Amazon EC2 mediante un SDK AWS](#). En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **DescribeInternetGateways** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `DescribeInternetGateways`.

### CLI

#### AWS CLI

Para describir una puerta de enlace a Internet

El siguiente `describe-internet-gateways` ejemplo describe la puerta de enlace de Internet especificada.

```
aws ec2 describe-internet-gateways \  
  --internet-gateway-ids igw-0d0fb496b3EXAMPLE
```

Salida:

```
{  
  "InternetGateways": [  
    {  
      "Attachments": [  
        {  
          "State": "available",  
          "VpcId": "vpc-0a60eb65b4EXAMPLE"  
        }  
      ],  
      "InternetGatewayId": "igw-0d0fb496b3EXAMPLE",  
      "OwnerId": "123456789012",  
      "Tags": [  
        {
```

```

        "Key": "Name",
        "Value": "my-igw"
      }
    ]
  }
}

```

Para obtener más información, consulte [Puertas de enlace de Internet](#) en la Guía del usuario de Amazon VPC.

- Para obtener más información sobre la API, consulte [DescribeInternetGateways](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: En este ejemplo se describe la puerta de enlace de Internet especificada.

```
Get-EC2InternetGateway -InternetGatewayId igw-1a2b3c4d
```

Salida:

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

Ejemplo 2: En este ejemplo se describen todas las puertas de enlace de Internet.

```
Get-EC2InternetGateway
```

Salida:

Attachments	InternetGatewayId	Tags
{vpc-1a2b3c4d}	igw-1a2b3c4d	{}
{}	igw-2a3b4c5d	{}

- Para obtener más información sobre la API, consulte la referencia [DescribeInternetGateways](#) de AWS Tools for PowerShell cmdlets.



Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **DescribeKeyPairs** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `DescribeKeyPairs`.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en contexto en el siguiente ejemplo de código:

- [Comience a utilizar instancias](#)

.NET

AWS SDK for .NET

### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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<KeyValuePairInfo>> DescribeKeyPairs(string keyPairName)
{
    var request = new DescribeKeyPairsRequest();
    if (!string.IsNullOrEmpty(keyPairName))
    {
        request = new DescribeKeyPairsRequest
        {
            KeyNames = new List<string> { keyPairName }
        };
    }
    var response = await _amazonEC2.DescribeKeyPairsAsync(request);
    return response.KeyPairs.ToList();
}
```

- Para obtener más información sobre la API, consulta [DescribeKeyPairs](#) la Referencia AWS SDK for .NET de la API.

## Bash

### AWS CLI con el script Bash

#### Note

Hay más información. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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
}

```

Las funciones de utilidad utilizadas en este ejemplo.

```

#####
# 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 obtener más información sobre la API, consulta [DescribeKeyPairs](#) la Referencia de AWS CLI comandos.

## C++

### SDK para C++

#### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

```
Aws::EC2::EC2Client ec2Client(clientConfiguration);
Aws::EC2::Model::DescribeKeyPairsRequest request;

auto outcome = ec2Client.DescribeKeyPairs(request);
if (outcome.IsSuccess()) {
    std::cout << std::left <<
                std::setw(32) << "Name" <<
                std::setw(64) << "Fingerprint" << std::endl;

    const std::vector<Aws::EC2::Model::KeyPairInfo> &key_pairs =
        outcome.GetResult().GetKeyPairs();
    for (const auto &key_pair: key_pairs) {
        std::cout << std::left <<
                    std::setw(32) << key_pair.GetKeyName() <<
                    std::setw(64) << key_pair.GetKeyFingerprint() << std::endl;
    }
}
else {
    std::cerr << "Failed to describe key pairs:" <<
              outcome.GetError().GetMessage() << std::endl;
}
```

- Para obtener más información sobre la API, consulta [DescribeKeyPairs](#) la Referencia AWS SDK for C++ de la API.

## CLI

### AWS CLI

#### Mostrar un par de claves

En el siguiente ejemplo de `describe-key-pairs`, se muestra información sobre el par de claves especificado.

```
aws ec2 describe-key-pairs \  
  --key-names my-key-pair
```

Salida:

```
{  
  "KeyPairs": [  
    {  
      "KeyId": "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 obtener más información, consulte [Describir las claves públicas](#) en la Guía del usuario de Amazon EC2.

- Para obtener más información sobre la API, consulta [DescribeKeyPairs](#) la Referencia de AWS CLI comandos.

## Java

### SDK para Java 2.x

#### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

```
public static void describeKeys(Ec2Client ec2) {  
  try {  
    DescribeKeyPairsResponse response = ec2.describeKeyPairs();
```

```
        response.keyPairs().forEach(keyPair -> System.out.printf(
            "Found key pair with name %s " +
            "and fingerprint %s",
            keyPair.keyName(),
            keyPair.keyFingerprint()));

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

- Para obtener más información sobre la API, consulta [DescribeKeyPairs](#) la Referencia AWS SDK for Java 2.x de la API.

## JavaScript

### SDK para JavaScript (v3)

#### Note

Hay más información. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

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

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

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

    try {
        const { KeyPairs } = await client.send(command);
        const keyPairList = KeyPairs.map(
            (kp) => ` • ${kp.KeyPairId}: ${kp.KeyName}`,
        ).join("\n");
        console.log("The following key pairs were found in your account:");
        console.log(keyPairList);
    } catch (err) {
```

```
    console.error(err);
  }
};
```

- Para obtener más información sobre la API, consulta [DescribeKeyPairs](#) la Referencia AWS SDK for JavaScript de la API.

## Kotlin

### SDK para Kotlin

#### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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 obtener más información sobre la API, consulta [DescribeKeyPairs](#) la referencia sobre el AWS SDK para la API de Kotlin.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: En este ejemplo se describe el key pair especificado.

```
Get-EC2KeyPair -KeyName my-key-pair
```



**Salida:**

```


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

```

Ejemplo 2: En este ejemplo se describen todos los pares de claves.

```
Get-EC2KeyPair
```

- Para obtener más información sobre la API, consulte [DescribeKeyPairs](#) la referencia de AWS Tools for PowerShell cmdlets.

**Python****SDK para Python (Boto3)**** Note**

Hay más información al respecto. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

```

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

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

```

```
self.key_pair = key_pair
self.key_file_path = None
self.key_file_dir = key_file_dir

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

def list(self, limit):
    """
    Displays a list of key pairs for the current account.

    :param limit: The maximum number of key pairs to list.
    """
    try:
        for kp in self.ec2_resource.key_pairs.limit(limit):
            print(f"Found {kp.key_type} key {kp.name} with fingerprint:")
            print(f"\t{kp.key_fingerprint}")
    except ClientError as err:
        logger.error(
            "Couldn't list key pairs. Here's why: %s: %s",
            err.response["Error"]["Code"],
            err.response["Error"]["Message"],
        )
        raise
```

- Para obtener más información sobre la API, consulta [DescribeKeyPairs](#) la AWS Referencia de API de SDK for Python (Boto3).

## SAP ABAP

### SDK de SAP ABAP

#### Note

Hay más información al respecto. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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 obtener más información sobre la API, consulte [DescribeKeyPairs](#) la referencia sobre la API ABAP del AWS SDK para SAP.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte [Cree recursos de Amazon EC2 mediante un SDK AWS](#). En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **DescribeNetworkAcls** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar DescribeNetworkAcls.

### CLI

#### AWS CLI

Para describir las ACL de su red

El siguiente describe-network-acls ejemplo recupera detalles sobre las ACL de la red.

```
aws ec2 describe-network-acls
```

Salida:

```

{
  "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 obtener más información, consulte [ACL de red](#) en la Guía del usuario de AWS VPC.

- Para obtener más información sobre la API, consulte la Referencia [DescribeNetworkAcls](#) de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: En este ejemplo se describe la ACL de red especificada.

```
Get-EC2NetworkAcl -NetworkAclId acl-12345678
```

Salida:

```

Associations : {aclassoc-1a2b3c4d}
Entries      : {Amazon.EC2.Model.NetworkAclEntry,
               Amazon.EC2.Model.NetworkAclEntry}
IsDefault    : False
NetworkAclId : acl-12345678
Tags         : {Name}

```

```
VpcId      : vpc-12345678
```

Ejemplo 2: En este ejemplo se describen las reglas de la ACL de red especificada.

```
(Get-EC2NetworkAcl -NetworkAclId acl-12345678).Entries
```

Salida:

```
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
```

Ejemplo 3: En este ejemplo se describen todas las ACL de la red.

```
Get-EC2NetworkAcl
```

- Para obtener información sobre la API, consulte la referencia [DescribeNetworkAcls](#) de AWS Tools for PowerShell cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Úselo **DescribeNetworkInterfaceAttribute** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `DescribeNetworkInterfaceAttribute`.

## CLI

### AWS CLI

Para describir el atributo adjunto de una interfaz de red

Este comando de ejemplo describe el `attachment` atributo de la interfaz de red especificada.

Comando:

```
aws ec2 describe-network-interface-attribute --network-interface-id eni-686ea200
--attribute attachment
```

Salida:

```
{
  "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 describir el atributo de descripción de una interfaz de red

Este comando de ejemplo describe el `description` atributo de la interfaz de red especificada.

Comando:

```
aws ec2 describe-network-interface-attribute --network-interface-id eni-686ea200
--attribute description
```

Salida:

```
{
  "NetworkInterfaceId": "eni-686ea200",
```



```
"Description": {
  "Value": "My description"
}
}
```

Para describir el atributo `groupSet` de una interfaz de red

Este comando de ejemplo describe el `groupSet` atributo de la interfaz de red especificada.

Comando:

```
aws ec2 describe-network-interface-attribute --network-interface-id eni-686ea200
--attribute groupSet
```

Salida:

```
{
  "NetworkInterfaceId": "eni-686ea200",
  "Groups": [
    {
      "GroupName": "my-security-group",
      "GroupId": "sg-903004f8"
    }
  ]
}
```

Para describir el `sourceDestCheck` atributo de una interfaz de red

Este comando de ejemplo describe el `sourceDestCheck` atributo de la interfaz de red especificada.

Comando:

```
aws ec2 describe-network-interface-attribute --network-interface-id eni-686ea200
--attribute sourceDestCheck
```

Salida:

```
{
  "NetworkInterfaceId": "eni-686ea200",
  "SourceDestCheck": {
```

```
    "Value": true
  }
}
```

- Para obtener más información sobre la API, consulte [DescribeNetworkInterfaceAttribute](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: en este ejemplo se describe la interfaz de red especificada.

```
Get-EC2NetworkInterfaceAttribute -NetworkInterfaceId eni-12345678 -Attribute Attachment
```

Salida:

```
Attachment           : Amazon.EC2.Model.NetworkInterfaceAttachment
```

Ejemplo 2: en este ejemplo se describe la interfaz de red especificada.

```
Get-EC2NetworkInterfaceAttribute -NetworkInterfaceId eni-12345678 -Attribute Description
```

Salida:

```
Description          : My description
```

Ejemplo 3: en este ejemplo se describe la interfaz de red especificada.

```
Get-EC2NetworkInterfaceAttribute -NetworkInterfaceId eni-12345678 -Attribute GroupSet
```

Salida:

```
Groups               : {my-security-group}
```

Ejemplo 4: En este ejemplo se describe la interfaz de red especificada.

```
Get-EC2NetworkInterfaceAttribute -NetworkInterfaceId eni-12345678 -Attribute
SourceDestCheck
```

Salida:

```
SourceDestCheck      : True
```

- Para obtener más información sobre la API, consulte [DescribeNetworkInterfaceAttribute](#) la referencia del AWS Tools for PowerShell cmdlet.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **DescribeNetworkInterfaces** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar DescribeNetworkInterfaces.

CLI

AWS CLI

Para describir las interfaces de red

En este ejemplo se describen todas las interfaces de red.

Comando:

```
aws ec2 describe-network-interfaces
```

Salida:

```
{
  "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"
}
```

```
]
}
```

En este ejemplo, se describen las interfaces de red que tienen una etiqueta con la clave Purpose y el valor Prod.

Comando:

```
aws ec2 describe-network-interfaces --filters Name=tag:Purpose,Values=Prod
```

Salida:

```
{
  "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 obtener más información sobre la API, consulte [DescribeNetworkInterfaces](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: en este ejemplo se describe la interfaz de red especificada.

```
Get-EC2NetworkInterface -NetworkInterfaceId eni-12345678
```

Salida:

```
Association      :
Attachment       : Amazon.EC2.Model.NetworkInterfaceAttachment
AvailabilityZone  : us-west-2c
Description      :
Groups           : {my-security-group}
MacAddress       : 0a:e9:a6:19:4c:7f
NetworkInterfaceId : eni-12345678
OwnerId          : 123456789012
PrivateDnsName   : ip-10-0-0-107.us-west-2.compute.internal
PrivateIpAddress : 10.0.0.107
PrivateIpAddresses : {ip-10-0-0-107.us-west-2.compute.internal}
RequesterId      :
RequesterManaged : False
SourceDestCheck  : True
Status           : in-use
SubnetId         : subnet-1a2b3c4d
```

```
TagSet      : {}  
VpcId       : vpc-12345678
```

Ejemplo 2: en este ejemplo se describen todas las interfaces de red.

```
Get-EC2NetworkInterface
```

- Para obtener más información sobre la API, consulte [DescribeNetworkInterfaces](#) la referencia de AWS Tools for PowerShell cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **DescribePlacementGroups** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar DescribePlacementGroups.

### CLI

#### AWS CLI

Para describir sus grupos de ubicación

Este comando de ejemplo describe todos los grupos de ubicación.

Comando:

```
aws ec2 describe-placement-groups
```

Salida:

```
{  
  "PlacementGroups": [  
    {  
      "GroupName": "my-cluster",  
      "State": "available",  
      "Strategy": "cluster"  
    },  
    ...  
  ]  
}
```



```
]
}
```

- Para obtener más información sobre la API, consulte [DescribePlacementGroups](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: en este ejemplo se describe el grupo de ubicación especificado.

```
Get-EC2PlacementGroup -GroupName my-placement-group
```

Salida:

GroupName	State	Strategy
-----	-----	-----
my-placement-group	available	cluster

- Para obtener más información sobre la API, consulte [DescribePlacementGroups](#) la referencia del AWS Tools for PowerShell cmdlet.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **DescribePrefixLists** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `DescribePrefixLists`.

### CLI

#### AWS CLI

Para describir las listas de prefijos

En este ejemplo se enumeran todas las listas de prefijos disponibles para la región.

Comando:

```
aws ec2 describe-prefix-lists
```

Salida:

```
{
  "PrefixLists": [
    {
      "PrefixListName": "com.amazonaws.us-east-1.s3",
      "Cidrs": [
        "54.231.0.0/17"
      ],
      "PrefixListId": "pl-63a5400a"
    }
  ]
}
```

- Para obtener más información sobre la API, consulte [DescribePrefixLists](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: En este ejemplo se busca lo disponible Servicios de AWS en un formato de lista de prefijos para la región

```
Get-EC2PrefixList
```

Salida:

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 obtener más información sobre la API, consulte la referencia de [DescribePrefixLists](#) [AWS Tools for PowerShell](#) cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **DescribeRegions** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar DescribeRegions.

C++

SDK para C++

### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

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

Aws::EC2::Model::DescribeRegionsRequest request;
auto outcome = ec2Client.DescribeRegions(request);
bool result = true;
if (outcome.IsSuccess()) {
    std::cout << std::left <<
        std::setw(32) << "RegionName" <<
        std::setw(64) << "Endpoint" << std::endl;

    const auto &regions = outcome.GetResult().GetRegions();
    for (const auto &region: regions) {
        std::cout << std::left <<
            std::setw(32) << region.GetRegionName() <<
            std::setw(64) << region.GetEndpoint() << std::endl;
    }
}
else {
    std::cerr << "Failed to describe regions:" <<
        outcome.GetError().GetMessage() << std::endl;
    result = false;
}
```

- Para obtener más información sobre la API, consulta [DescribeRegions](#) la Referencia AWS SDK for C++ de la API.

## CLI

### AWS CLI

#### Ejemplo 1: Describir todas las regiones habilitadas

En el siguiente ejemplo de `describe-regions`, se describen las regiones que están habilitadas para su cuenta.

```
aws ec2 describe-regions
```

#### Salida:

```
{
  "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 obtener más información, consulte [Regiones y zonas](#) en la Guía del usuario de Amazon EC2.

Ejemplo 2: Describir las regiones habilitadas con un punto de conexión cuyo nombre contiene una cadena específica

En el siguiente ejemplo de `describe-regions`, se describen todas las regiones que ha habilitado y que tienen la cadena “us” en el punto de conexión.

```

aws ec2 describe-regions \
  --filters "Name=endpoint,Values=*us*"

```

Salida:

```

{
  "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 obtener más información, consulte [Regiones y zonas](#) en la Guía del usuario de Amazon EC2.

### Ejemplo 3: Describir todas las regiones

En el siguiente ejemplo de `describe-regions`, se describen todas las regiones disponibles, incluidas las que están deshabilitadas.

```
aws ec2 describe-regions \
  --all-regions
```

Salida:

```
{
  "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 obtener más información, consulte [Regiones y zonas](#) en la Guía del usuario de Amazon EC2.

Ejemplo 4: Enumerar únicamente los nombres de las regiones

En el siguiente ejemplo de `describe-regions`, se usa el parámetro `--query` para filtrar la salida y devolver solo los nombres de las regiones como texto.

```
aws ec2 describe-regions \  
  --all-regions \  
  --query "Regions[].{Name:RegionName}" \  
  --output text
```

Salida:

```
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 obtener más información, consulte [Regiones y zonas](#) en la Guía del usuario de Amazon EC2.

- Para obtener más información sobre la API, consulta [DescribeRegions](#) la Referencia de AWS CLI comandos.

## JavaScript

### SDK para JavaScript (v3)

#### Note

Hay más información. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

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

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

export const main = async () => {
  const command = new DescribeRegionsCommand({
    // By default this command will not show regions that require you to opt-in.
    // When AllRegions true even the regions that require opt-in will be
    returned.
    AllRegions: true,
    // You can omit the Filters property if you want to get all regions.
    Filters: [
      {
        Name: "region-name",
        // You can specify multiple values for a filter.
        // You can also use '*' as a wildcard. This will return all
        // of the regions that start with `us-east-`.
        Values: ["ap-southeast-4"],
      },
    ],
  });

  try {
    const { Regions } = await client.send(command);
    const regionsList = Regions.map((reg) => ` • ${reg.RegionName}`);
    console.log("Found regions:");
    console.log(regionsList.join("\n"));
  } catch (err) {
    console.error(err);
  }
};
```

- Para obtener más información sobre la API, consulta [DescribeRegions](#) la Referencia AWS SDK for JavaScript de la API.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: En este ejemplo se describen las regiones que están disponibles para usted.

```
Get-EC2Region
```

Salida:

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 obtener información sobre la API, consulte [DescribeRegions](#) la referencia de AWS Tools for PowerShell cmdlets.

## Ruby

### SDK para Ruby

#### Note

Hay más información al respecto. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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 obtener más información sobre la API, consulta [DescribeRegions](#) la Referencia AWS SDK for Ruby de la API.

## Rust

### SDK para Rust

#### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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 obtener más información sobre la API, consulta [DescribeRegions](#) la referencia sobre la API de AWS SDK para Rust.

## SAP ABAP

### SDK de SAP ABAP

#### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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 obtener más información sobre la API, consulte [DescribeRegions](#) la referencia sobre la API ABAP del AWS SDK para SAP.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte [Cree recursos de Amazon EC2 mediante un SDK AWS](#). En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **DescribeRouteTables** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar DescribeRouteTables.



## CLI

### AWS CLI

Para describir sus tablas de rutas

En el siguiente `describe-route-tables` ejemplo, se recuperan los detalles de las tablas de rutas

```
aws ec2 describe-route-tables
```

Salida:

```
{
  "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 obtener más información, consulte [Trabajar con tablas de enrutamiento](#) en la Guía del usuario de la AWS VPC.

- Para obtener más información sobre la API, consulte [DescribeRouteTables](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: En este ejemplo se describen todas las tablas de rutas.

```
Get-EC2RouteTable
```

Salida:

```

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 :

```

Ejemplo 2: Este ejemplo devuelve los detalles de la tabla de rutas especificada.

```
Get-EC2RouteTable -RouteTableId rtb-1a2b3c4d
```

Ejemplo 3: en este ejemplo se describen las tablas de enrutamiento de la VPC especificada.

```
Get-EC2RouteTable -Filter @{ Name="vpc-id"; Values="vpc-1a2b3c4d" }
```

Salida:

```

Associations    : {rtbassoc-12345678}
PropagatingVgws : {}
Routes          : {, }
RouteTableId    : rtb-1a2b3c4d
Tags            : {}
VpcId           : vpc-1a2b3c4d

```

- Para obtener información sobre la API, consulte la referencia [DescribeRouteTables](#) de AWS Tools for PowerShell cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **DescribeScheduledInstanceAvailability** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `DescribeScheduledInstanceAvailability`.

## CLI

## AWS CLI

Para describir un horario disponible

En este ejemplo, se describe un horario que se realiza todos los domingos a partir de la fecha 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
```

Salida:

```
{
  "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 limitar los resultados, puede agregar filtros que especifiquen el sistema operativo, la red y el tipo de instancia.

Comando:

```
--filters name=Plataforma, values=Linux/UNIX Name=Network-Platform, values=EC2-VPC
Name=Instance-Type, values=C4.large
```

- Para obtener más información sobre la API, consulte la Referencia de comandos [DescribeScheduledInstanceAvailability](#).AWS CLI

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: en este ejemplo se describe un horario que se realiza todos los domingos a partir de la fecha 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
```

Salida:

```
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
```

...

Ejemplo 2: Para limitar los resultados, puede añadir filtros para criterios como el sistema operativo, la red y el tipo de instancia.

```
-Filter @{ Name="platform";Values="Linux/UNIX" },@{ Name="network-  
platform";Values="EC2-VPC" },@{ Name="instance-type";Values="c4.large" }
```

- Para obtener más información sobre la API, consulte [DescribeScheduledInstanceAvailability](#) la referencia de AWS Tools for PowerShell cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **DescribeScheduledInstances** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar DescribeScheduledInstances.

### CLI

#### AWS CLI

Para describir sus instancias programadas

En este ejemplo se describe la instancia programada especificada.

Comando:

```
aws ec2 describe-scheduled-instances --scheduled-instance-ids  
sci-1234-1234-1234-1234-123456789012
```

Salida:

```
{  
  "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"
  }
]
}

```

En este ejemplo se describen todas las instancias programadas.

Comando:

```
aws ec2 describe-scheduled-instances
```

- Para obtener información sobre la API, consulte [DescribeScheduledInstances](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: en este ejemplo se describe la instancia programada especificada.

```
Get-EC2ScheduledInstance -ScheduledInstanceId
sci-1234-1234-1234-1234-123456789012
```



**Salida:**

```
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
```

Ejemplo 2: En este ejemplo se describen todas las instancias programadas.

```
Get-EC2ScheduledInstance
```

- Para obtener más información sobre la API, consulte [DescribeScheduledInstances](#) la referencia de AWS Tools for PowerShell cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **DescribeSecurityGroups** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `DescribeSecurityGroups`.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en contexto en el siguiente ejemplo de código:

- [Comience a utilizar instancias](#)

## .NET

### AWS SDK for .NET

#### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

```
/// <summary>
/// Retrieve information for an Amazon EC2 security group.
/// </summary>
/// <param name="groupId">The Id of the Amazon EC2 security group.</param>
/// <returns>A list of security group information.</returns>
public async Task<List<SecurityGroup>> DescribeSecurityGroups(string groupId)
{
    var request = new DescribeSecurityGroupsRequest();
    var groupIds = new List<string> { groupId };
    request.GroupIds = groupIds;

    var response = await _amazonEC2.DescribeSecurityGroupsAsync(request);
    return response.SecurityGroups;
}

/// <summary>
/// Display the information returned by the call to
/// DescribeSecurityGroupsAsync.
/// </summary>
/// <param name="securityGroup">A list of security group information.</param>
public void DisplaySecurityGroupInfoAsync(SecurityGroup securityGroup)
{
    Console.WriteLine($"{securityGroup.GroupName}");
    Console.WriteLine("Ingress permissions:");
    securityGroup.IpPermissions.ForEach(permission =>
    {
        Console.WriteLine($"  \tFromPort: {permission.FromPort}");
        Console.WriteLine($"  \tIpProtocol: {permission.IpProtocol}");

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

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

        Console.Write($"\\n\\tPrefixListIds: ");
        permission.PrefixListIds.ForEach(id => Console.Write($"{id.Id} "));

        Console.WriteLine($"\\n\\tTo Port: {permission.ToPort}");
    });
    Console.WriteLine("Egress permissions:");
    securityGroup.IpPermissionsEgress.ForEach(permission =>
    {
        Console.WriteLine($"\\tFromPort: {permission.FromPort}");
        Console.WriteLine($"\\tIpProtocol: {permission.IpProtocol}");

        Console.Write($"\\tIpv4Ranges: ");
        permission.Ipv4Ranges.ForEach(range =>
{ Console.Write($"{range.CidrIp} "); });

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

        Console.Write($"\\n\\tPrefixListIds: ");
        permission.PrefixListIds.ForEach(id => Console.Write($"{id.Id} "));

        Console.WriteLine($"\\n\\tTo Port: {permission.ToPort}");
    });
}
```

- Para obtener más información sobre la API, consulta [DescribeSecurityGroups](#) la Referencia AWS SDK for .NET de la API.

## Bash

### AWS CLI con el script Bash

#### Note

Hay más información. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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
}

```

Las funciones de utilidad utilizadas en este ejemplo.

```

#####
# 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 obtener más información sobre la API, consulta [DescribeSecurityGroups](#) la Referencia de AWS CLI comandos.

## C++

## SDK para C++

 Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

```
Aws::EC2::EC2Client ec2Client(clientConfiguration);
Aws::EC2::Model::DescribeSecurityGroupsRequest request;

if (!groupID.empty()) {
    request.AddGroupIds(groupID);
}

Aws::String nextToken;
do {
    if (!nextToken.empty()) {
        request.SetNextToken(nextToken);
    }

    auto outcome = ec2Client.DescribeSecurityGroups(request);
    if (outcome.IsSuccess()) {
        std::cout << std::left <<
            std::setw(32) << "Name" <<
            std::setw(30) << "GroupId" <<
            std::setw(30) << "VpcId" <<
            std::setw(64) << "Description" << std::endl;

        const std::vector<Aws::EC2::Model::SecurityGroup> &securityGroups =
            outcome.GetResult().GetSecurityGroups();

        for (const auto &securityGroup: securityGroups) {
            std::cout << std::left <<
                std::setw(32) << securityGroup.GetGroupName() <<
                std::setw(30) << securityGroup.GetGroupId() <<
                std::setw(30) << securityGroup.GetVpcId() <<
                std::setw(64) << securityGroup.GetDescription() <<
                std::endl;
        }
    }
}
```

```
    }
    else {
        std::cerr << "Failed to describe security groups:" <<
            outcome.GetError().GetMessage() << std::endl;
        return false;
    }

    nextToken = outcome.GetResult().GetNextToken();
} while (!nextToken.empty());
```

- Para obtener más información sobre la API, consulta [DescribeSecurityGroups](#) la Referencia AWS SDK for C++ de la API.

## CLI

### AWS CLI

#### Ejemplo 1: Describir un grupo de seguridad

En el siguiente ejemplo de `describe-security-groups`, se describe el grupo de seguridad especificado.

```
aws ec2 describe-security-groups \
  --group-ids sg-903004f8
```

#### Salida:

```
{
  "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",
  }
]
}

```

## Ejemplo 2: Describir los grupos de seguridad que tienen reglas específicas

En el siguiente `describe-security-groups` ejemplo, se utilizan filtros para asignar los resultados a los grupos de seguridad que tienen una regla que permite el tráfico SSH (puerto 22) y una regla que permite el tráfico desde todas las direcciones (`0.0.0.0/0`). En el ejemplo se usa el parámetro `--query` para mostrar solo los nombres de los grupos de seguridad. Los grupos de seguridad deben coincidir con todos los filtros para que se devuelvan en los resultados; sin embargo, una sola regla no tiene que coincidir con todos los filtros. Por ejemplo, el resultado devuelve un grupo de seguridad con una regla que permite el tráfico SSH desde una dirección IP específica y otra regla que permite el tráfico HTTP desde todas las direcciones.

```
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
```

Salida:

```
default
my-security-group
web-servers
launch-wizard-1
```

Ejemplo 3: Describir los grupos de seguridad con base en las etiquetas

En el siguiente ejemplo de `describe-security-groups`, se usan filtros para limitar los resultados a los grupos de seguridad que incluyen `test` en el nombre del grupo de seguridad y que tienen la etiqueta `Test=To-delete`. En el ejemplo se usa el parámetro `--query` para mostrar solo los nombres y los ID de los grupos de seguridad.

```
aws ec2 describe-security-groups \
  --filters Name=group-name,Values=*test* Name=tag:Test,Values=To-delete \
  --query "SecurityGroups[*].{Name:GroupName, ID:GroupId}"
```

Salida:

```
[
  {
    "Name": "testfornewinstance",
```

```
    "ID": "sg-33bb22aa"
  },
  {
    "Name": "newgroupptest",
    "ID": "sg-1a2b3c4d"
  }
]
```

Para ver ejemplos adicionales de uso de filtros de etiquetas, consulte [Trabajo con etiquetas](#) en la Guía del usuario de Amazon EC2.

- Para obtener más información sobre la API, consulte [DescribeSecurityGroups](#) la Referencia de AWS CLI comandos.

## Java

### SDK para Java 2.x

#### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

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

        // Use a paginator.
        DescribeSecurityGroupsIterable listGroups =
ec2.describeSecurityGroupsPaginator(request);
        listGroups.stream()
            .flatMap(r -> r.securityGroups().stream())
            .forEach(group -> System.out
                .println(" Group id: " +group.groupId() + " group name = " +
group.groupName()));
    } catch (Ec2Exception e) {
```

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

- Para obtener más información sobre la API, consulta [DescribeSecurityGroups](#) la Referencia AWS SDK for Java 2.x de la API.

## JavaScript

### SDK para JavaScript (v3)

#### Note

Hay más información. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

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

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

// Log the details of a specific security group.
export const main = async () => {
    const command = new DescribeSecurityGroupsCommand({
        GroupIds: ["SECURITY_GROUP_ID"],
    });

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

- Para obtener más información sobre la API, consulta [DescribeSecurityGroups](#) la Referencia AWS SDK for JavaScript de la API.

## Kotlin

### SDK para Kotlin

#### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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 obtener más información sobre la API, consulta [DescribeSecurityGroups](#) la referencia sobre el AWS SDK para la API de Kotlin.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: en este ejemplo se describe el grupo de seguridad especificado para una VPC. Al trabajar con grupos de seguridad que pertenecen a una VPC, debe usar el ID del grupo de seguridad (GroupId parámetro -), no el nombre (GroupName parámetro -), para hacer referencia al grupo.

```
Get-EC2SecurityGroup -GroupId sg-12345678
```

**Salida:**

```
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
```

Ejemplo 2: en este ejemplo se describe el grupo de seguridad especificado para EC2-Classic. Al trabajar con grupos de seguridad para EC2-Classic, puede utilizar el nombre del grupo (GroupName parámetro -) o el ID del grupo (GroupId parámetro -) para hacer referencia al grupo de seguridad.

```
Get-EC2SecurityGroup -GroupName my-security-group
```

**Salida:**

```
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           :
```

Ejemplo 3: en este ejemplo se recuperan todos los grupos de seguridad del vpc-0fc1ff23456b789eb

```
Get-EC2SecurityGroup -Filter @{Name="vpc-id";Values="vpc-0fc1ff23456b789eb"}
```

- Para [DescribeSecurityGroups](#) obtener más información sobre AWS Tools for PowerShell la API, consulte la referencia del cmdlet.

## Python

### SDK para Python (Boto3)

#### Note

Hay más información al respecto. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

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

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

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

    def describe(self):
        """
        Displays information about the security group.
        """
        if self.security_group is None:
            logger.info("No security group to describe.")
            return

        try:
```

```

print(f"Security group: {self.security_group.group_name}")
print(f"\tID: {self.security_group.id}")
print(f"\tVPC: {self.security_group.vpc_id}")
if self.security_group.ip_permissions:
    print(f"Inbound permissions:")
    pp(self.security_group.ip_permissions)
except ClientError as err:
    logger.error(
        "Couldn't get data for security group %s. Here's why: %s: %s",
        self.security_group.id,
        err.response["Error"]["Code"],
        err.response["Error"]["Message"],
    )
    raise

```

- Para obtener más información sobre la API, consulta [DescribeSecurityGroups](#) la AWS Referencia de API de SDK for Python (Boto3).

## SAP ABAP

### SDK de SAP ABAP

#### Note

Hay más información al respecto. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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 obtener más información sobre la API, consulte [DescribeSecurityGroups](#) la referencia sobre la API ABAP del AWS SDK para SAP.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte [Cree recursos de Amazon EC2 mediante un SDK AWS](#). En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **DescribeSnapshotAttribute** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `DescribeSnapshotAttribute`.

### CLI

#### AWS CLI

Para describir los atributos de una instantánea

En el siguiente `describe-snapshot-attribute` ejemplo, se enumeran las cuentas con las que se comparte una instantánea.

```
aws ec2 describe-snapshot-attribute \  
  --snapshot-id snap-01234567890abcdef \  
  --attribute createVolumePermission
```

Salida:

```
{  
  "SnapshotId": "snap-01234567890abcdef",  
  "CreateVolumePermissions": [  
    {  
      "UserId": "123456789012"  
    }  
  ]  
}
```

Para obtener más información, consulte [Compartir una instantánea de Amazon EBS](#) en la Guía del usuario de Amazon Elastic Compute Cloud.

- Para obtener más información sobre la API, consulte [DescribeSnapshotAttribute](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: en este ejemplo se describe el atributo especificado de la instantánea especificada.

```
Get-EC2SnapshotAttribute -SnapshotId snap-12345678 -Attribute ProductCodes
```

Salida:

CreateVolumePermissions	ProductCodes	SnapshotId
-----	-----	-----
{}	{}	snap-12345678

Ejemplo 2: en este ejemplo se describe el atributo especificado de la instantánea especificada.

```
(Get-EC2SnapshotAttribute -SnapshotId snap-12345678 -Attribute
CreateVolumePermission).CreateVolumePermissions
```

Salida:

Group	UserId
-----	-----
all	

- Para obtener más información sobre la API, consulte [DescribeSnapshotAttribute](#) la referencia del AWS Tools for PowerShell cmdlet.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **DescribeSnapshots** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar DescribeSnapshots.

### CLI

#### AWS CLI

##### Ejemplo 1: Describir una instantánea

En el siguiente ejemplo de describe-snapshots, se describe la instantánea especificada.

```
aws ec2 describe-snapshots \
  --snapshot-ids snap-1234567890abcdef0
```

#### Salida:

```
{
  "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 obtener más información, consulte [Instantáneas de Amazon EBS](#) en la Guía del usuario de Amazon EC2.

## Ejemplo 2: Describir las instantáneas con base en filtros

En el siguiente `describe-snapshots` ejemplo, se utilizan filtros para limitar los resultados a las instantáneas que son propiedad de su AWS cuenta y que se encuentran en el `pending` estado. En el ejemplo se usa el parámetro `--query` para mostrar solo los ID de las instantáneas y la hora en que se inició la instantánea.

```
aws ec2 describe-snapshots \  
  --owner-ids self \  
  --filters Name=status,Values=pending \  
  --query "Snapshots[*].{ID:SnapshotId,Time:StartTime}"
```

Salida:

```
[  
  {  
    "ID": "snap-1234567890abcdef0",  
    "Time": "2019-08-04T12:48:18.000Z"  
  },  
  {  
    "ID": "snap-066877671789bd71b",  
    "Time": "2019-08-04T02:45:16.000Z"  
  },  
  ...  
]
```

En el siguiente ejemplo de `describe-snapshots`, se usan filtros para limitar los resultados a las instantáneas creadas a partir del volumen especificado. En el ejemplo se usa el parámetro `--query` para mostrar solo los ID de las instantáneas.

```
aws ec2 describe-snapshots \  
  --filters Name=volume-id,Values=049df61146c4d7901 \  
  --query "Snapshots[*].[SnapshotId]" \  
  --output text
```

Salida:

```
snap-1234567890abcdef0  
snap-08637175a712c3fb9  
...
```

Para ver ejemplos adicionales de uso de filtros, consulte [Enumerar y filtrar los recursos](#) en la Guía del usuario de Amazon EC2.

### Ejemplo 3: Describir las instantáneas con base en etiquetas

En el siguiente ejemplo de `describe-snapshots`, se usan filtros de etiquetas para limitar los resultados a las instantáneas que tienen la etiqueta `Stack=Prod`.

```
aws ec2 describe-snapshots \  
  --filters Name=tag:Stack,Values=prod
```

Para ver un ejemplo del resultado de `describe-snapshots`, consulte el ejemplo 1.

Para ver ejemplos adicionales de uso de filtros de etiquetas, consulte [Trabajo con etiquetas](#) en la Guía del usuario de Amazon EC2.

### Ejemplo 4: Describir las instantáneas con base en la antigüedad

En el siguiente `describe-snapshots` ejemplo, se utilizan expresiones de JMESpath para describir todas las instantáneas creadas por la AWS cuenta antes de la fecha especificada. Solo se muestran los ID de las instantáneas.

```
aws ec2 describe-snapshots \  
  --owner-ids 012345678910 \  
  --query "Snapshots[?(StartTime<='2020-03-31')].[SnapshotId]"
```

Para ver ejemplos adicionales de uso de filtros, consulte [Enumerar y filtrar los recursos](#) en la Guía del usuario de Amazon EC2.

### Ejemplo 5: Ver solo las instantáneas archivadas

En el siguiente ejemplo de `describe-snapshots`, se muestran solo las instantáneas que se almacenan en el nivel de archivo.

```
aws ec2 describe-snapshots \  
  --filters "Name=storage-tier,Values=archive"
```

Salida:

```
{  
  "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 obtener más información, consulte [Ver instantáneas archivadas](#) en la Guía del usuario de Amazon Elastic Compute Cloud.

- Para obtener más información sobre la API, consulte [DescribeSnapshots](#) la Referencia de comandos.AWS CLI

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: en este ejemplo se describe la instantánea especificada.

```
Get-EC2Snapshot -SnapshotId snap-12345678
```

### Salida:

```
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

```

Ejemplo 2: en este ejemplo se describen las instantáneas que tienen la etiqueta «Nombre».

```
Get-EC2Snapshot | ? { $_.Tags.Count -gt 0 -and $_.Tags.Key -eq "Name" }
```

Ejemplo 3: En este ejemplo se describen las instantáneas que tienen una etiqueta de «Nombre» con el valor «». TestValue

```
Get-EC2Snapshot | ? { $_.Tags.Count -gt 0 -and $_.Tags.Key -eq "Name" -and
  $_.Tags.Value -eq "TestValue" }
```

Ejemplo 4: En este ejemplo se describen todas las instantáneas.

```
Get-EC2Snapshot -Owner self
```

- Para obtener más información sobre la API, consulte la referencia [DescribeSnapshots](#) de AWS Tools for PowerShell cmdlets.

## Rust

### SDK para Rust

#### Note

Hay más información al respecto. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

Muestra el estado de una instantánea.

```

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 obtener más información sobre la API, consulta [DescribeSnapshots](#) la referencia sobre la API de AWS SDK para Rust.



Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte [Cree recursos de Amazon EC2 mediante un SDK AWS](#). En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo `DescribeSpotDatafeedSubscription` con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `DescribeSpotDatafeedSubscription`.

### CLI

#### AWS CLI

Para describir la suscripción a la fuente de datos de Spot Instance para una cuenta

Este comando de ejemplo describe la fuente de datos de la cuenta.

Comando:

```
aws ec2 describe-spot-datafeed-subscription
```

Salida:

```
{
  "SpotDatafeedSubscription": {
    "OwnerId": "123456789012",
    "Prefix": "spotdata",
    "Bucket": "my-s3-bucket",
    "State": "Active"
  }
}
```

- Para obtener más información sobre la API, consulte [DescribeSpotDatafeedSubscription](#) la Referencia de AWS CLI comandos.

### PowerShell

#### Herramientas para PowerShell

Ejemplo 1: En este ejemplo se describe la fuente de datos de su instancia de Spot.

```
Get-EC2SpotDatafeedSubscription
```

Salida:

```
Bucket   : my-s3-bucket
Fault    :
OwnerId  : 123456789012
Prefix   : spotdata
State    : Active
```

- Para obtener más información sobre la API, consulte [DescribeSpotDatafeedSubscription](#) la referencia de AWS Tools for PowerShell cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **DescribeSpotFleetInstances** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar DescribeSpotFleetInstances.

CLI

AWS CLI

Para describir las instancias puntuales asociadas a una flota de Spot

Este comando de ejemplo muestra las instancias de Spot asociadas a la flota de Spot especificada.

Comando:

```
aws ec2 describe-spot-fleet-instances --spot-fleet-request-id sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE
```

Salida:

```
{
  "ActiveInstances": [
    {
```

```

        "InstanceId": "i-1234567890abcdef0",
        "InstanceType": "m3.medium",
        "SpotInstanceRequestId": "sir-08b93456"
    },
    ...
  ],
  "SpotFleetRequestId": "sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE"
}

```

- Para obtener más información sobre la API, consulte [DescribeSpotFleetInstances](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: En este ejemplo se describen las instancias asociadas a la solicitud de flota de Spot especificada.

```
Get-EC2SpotFleetInstance -SpotFleetRequestId sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE
```

Salida:

InstanceId	InstanceType	SpotInstanceRequestId
-----	-----	-----
i-f089262a	c3.large	sir-12345678
i-7e8b24a4	c3.large	sir-87654321

- Para obtener más información sobre la API, consulte [DescribeSpotFleetInstances](#) la referencia de AWS Tools for PowerShell cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Úselo **DescribeSpotFleetRequestHistory** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `DescribeSpotFleetRequestHistory`.

## CLI

### AWS CLI

Para describir el historial de la flota de Spot

Este comando de ejemplo devuelve el historial de la flota de Spot especificada a partir de la hora especificada.

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
```

En el siguiente ejemplo, se muestran los lanzamientos satisfactorios de dos instancias de spot para la flota de Spot.

Salida:

```
{
  "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
+53MNLR0YcXAkp0xF1fKf91yVxSExmbtma3awYxMFzNA663ZskT0AhtJ6TCb2Z8bQC2EnZgyELbymtWPfpZ1ZbauV
+P+TfG1WxWWB/Vr5dk5d4LfdgA/DRAHUrYgxzrEXAMPLE=",
  "StartTime": "2015-05-26T00:00:00Z"
}

```

- Para obtener más información sobre la API, consulte [DescribeSpotFleetRequestHistory](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: En este ejemplo se describe el historial de la solicitud de flota de Spot especificada.

```
Get-EC2SpotFleetRequestHistory -SpotFleetRequestId sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE -StartTime 2015-12-26T00:00:00Z
```

Salida:

```

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
```

**Salida:**

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 obtener más información sobre la API, consulte [DescribeSpotFleetRequestHistory](#) la referencia de AWS Tools for PowerShell cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **DescribeSpotFleetRequests** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar DescribeSpotFleetRequests.

**CLI****AWS CLI**

Para describir sus solicitudes de flota de Spot

En este ejemplo se describen todas sus solicitudes de flota de Spot.

Comando:

```
aws ec2 describe-spot-fleet-requests
```

Salida:

```
{
  "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 describir una solicitud de flota de Spot

En este ejemplo se describe la solicitud de flota de Spot especificada.

Comando:

```
aws ec2 describe-spot-fleet-requests --spot-fleet-request-ids sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE
```

Salida:

```

{
  "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 obtener más información sobre la API, consulte [DescribeSpotFleetRequests](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: En este ejemplo se describe la solicitud de flota de Spot especificada.

```
Get-EC2SpotFleetRequest -SpotFleetRequestId sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE | format-list
```

**Salida:**

```
ConfigData      : Amazon.EC2.Model.SpotFleetRequestConfigData
CreateTime     : 12/26/2015 8:23:33 AM
SpotFleetRequestId : sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE
SpotFleetRequestState : active
```

Ejemplo 2: En este ejemplo se describen todas las solicitudes de flota de Spot.

```
Get-EC2SpotFleetRequest
```

- Para obtener más información sobre la API, consulte [DescribeSpotFleetRequests](#) la referencia de AWS Tools for PowerShell cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **DescribeSpotInstanceRequests** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `DescribeSpotInstanceRequests`.

### CLI

#### AWS CLI

Ejemplo 1: Para describir una solicitud de instancia puntual

El siguiente `describe-spot-instance-requests` ejemplo describe la solicitud de instancia de spot especificada.

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

**Salida:**

```
{
  "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"
}
]
}

```

### Ejemplo 2: Para describir las solicitudes de instancias puntuales basadas en filtros

En el siguiente `describe-spot-instance-requests` ejemplo, se utilizan filtros para limitar los resultados a las solicitudes de instancias puntuales con el tipo de instancia especificado en la zona de disponibilidad especificada. En el ejemplo, se usa el `--query` parámetro para mostrar solo los ID de las instancias.

```

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

```

Salida:

```

i-057750d42936e468a
i-001efd250faaa6ffa
i-027552a73f021f3bd
...

```

Para ver ejemplos adicionales de uso de filtros, consulte [Cómo enumerar y filtrar los recursos](#) en la Guía del usuario de Amazon Elastic Compute Cloud.

### Ejemplo 3: Para describir las solicitudes de instancias puntuales basadas en etiquetas

En el siguiente `describe-spot-instance-requests` ejemplo, se utilizan filtros de etiquetas para limitar los resultados a las solicitudes de instancias puntuales que tienen la etiqueta `cost-center=cc123`.

```

aws ec2 describe-spot-instance-requests \

```

```
--filters Name=tag:cost-center,Values=cc123
```

Para ver un ejemplo del resultado de `describe-spot-instance-requests`, consulte el ejemplo 1.

Para ver ejemplos adicionales de uso de filtros de etiquetas, consulte [Trabajo con etiquetas](#) en la Guía del usuario de Amazon EC2.

- Para obtener más información sobre la API, consulte [DescribeSpotInstanceRequests](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: en este ejemplo se describe la solicitud de instancia de spot especificada.

```
Get-EC2SpotInstanceRequest -SpotInstanceRequestId sir-12345678
```

Salida:

```
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
```

Ejemplo 2: En este ejemplo se describen todas las solicitudes de instancias de spot.

```
Get-EC2SpotInstanceRequest
```

- Para obtener más información sobre la API, consulte [DescribeSpotInstanceRequests](#) la referencia de AWS Tools for PowerShell cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **DescribeSpotPriceHistory** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `DescribeSpotPriceHistory`.

### CLI

#### AWS CLI

Para describir el historial de precios al contado

Este comando de ejemplo devuelve el historial de precios al contado de las instancias `m1.xlarge` de un día concreto de enero.

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
```

Salida:

```
{
  "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 describir el historial de precios al contado de Amazon VPC para Linux/UNIX

Este comando de ejemplo devuelve el historial de precios puntuales de las instancias m1.xlarge de Amazon VPC Linux/UNIX de un día concreto de enero.

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

```

Salida:

```

{
  "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 obtener más información sobre la API, consulte la Referencia de comandos.

[DescribeSpotPriceHistory](#) AWS CLI

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: En este ejemplo se obtienen las últimas 10 entradas del historial de precios al contado para el tipo de instancia y la zona de disponibilidad especificados. Tenga en cuenta que el valor especificado para el AvailabilityZone parámetro - debe ser válido para el valor de región proporcionado al parámetro -Region del cmdlet (que no se muestra en el ejemplo) o estar establecido como predeterminado en el shell. Este comando de ejemplo supone que se ha establecido una región predeterminada de 'us-west-2' en el entorno.

```
Get-EC2SpotPriceHistory -InstanceType c3.large -AvailabilityZone us-west-2a -
MaxResult 10
```

### Salida:

```
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 obtener más información sobre la API, consulte la referencia del [DescribeSpotPriceHistory AWS Tools for PowerShell](#) cmdlet.



Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **DescribeSubnets** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar DescribeSubnets.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en contexto en el siguiente ejemplo de código:

- [Creación y administración de un servicio resiliente](#)

### .NET

#### AWS SDK for .NET

##### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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)
{
    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;
}
```

- Para obtener más información sobre la API, consulta [DescribeSubnets](#) la Referencia AWS SDK for .NET de la API.

## CLI

### AWS CLI

#### Ejemplo 1: Describir todas las subredes

En el siguiente ejemplo de `describe-subnets`, se muestran los detalles de las subredes.

```
aws ec2 describe-subnets
```

#### Salida:

```
{
  "Subnets": [
    {
      "AvailabilityZone": "us-east-1d",
      "AvailabilityZoneId": "use1-az2",
      "AvailableIpAddressCount": 4089,
      "CidrBlock": "172.31.80.0/20",
      "DefaultForAz": true,
      "MapPublicIpOnLaunch": false,
      "MapCustomerOwnedIpOnLaunch": true,
      "State": "available",
      "SubnetId": "subnet-0bb1c79de3EXAMPLE",
      "VpcId": "vpc-0ee975135dEXAMPLE",
      "OwnerId": "111122223333",
    }
  ]
}
```

```

    "AssignIpv6AddressOnCreation": false,
    "Ipv6CidrBlockAssociationSet": [],
    "CustomerOwnedIpv4Pool": "pool-2EXAMPLE",
    "SubnetArn": "arn:aws:ec2:us-east-2:111122223333:subnet/
subnet-0bb1c79de3EXAMPLE",
    "EnableDns64": false,
    "Ipv6Native": false,
    "PrivateDnsNameOptionsOnLaunch": {
      "HostnameType": "ip-name",
      "EnableResourceNameDnsARecord": false,
      "EnableResourceNameDnsAAAARecord": false
    }
  },
  {
    "AvailabilityZone": "us-east-1d",
    "AvailabilityZoneId": "use1-az2",
    "AvailableIpAddressCount": 4089,
    "CidrBlock": "172.31.80.0/20",
    "DefaultForAz": true,
    "MapPublicIpOnLaunch": true,
    "MapCustomerOwnedIpOnLaunch": false,
    "State": "available",
    "SubnetId": "subnet-8EXAMPLE",
    "VpcId": "vpc-3EXAMPLE",
    "OwnerId": "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 obtener más información, consulte [Trabajo con VPC y subredes](#) en la Guía del usuario de AWS VPC.

### Ejemplo 2: Describir las subredes de una VPC específica

En el ejemplo siguiente de `describe-subnets` se utiliza un filtro para recuperar detalles de las subredes de la VPC especificada.

```
aws ec2 describe-subnets \  
  --filters "Name=vpc-id,Values=vpc-3EXAMPLE"
```

Salida:

```
{  
  "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 obtener más información, consulte [Trabajo con VPC y subredes](#) en la Guía del usuario de AWS VPC.

### Ejemplo 3: Describir las subredes con una etiqueta específica

En el siguiente ejemplo de `describe-subnets`, se usa un filtro para recuperar los detalles de las subredes con la etiqueta `CostCenter=123` y el parámetro `--query` a fin de mostrar los ID de subred de las subredes con esta etiqueta.

```
aws ec2 describe-subnets \
  --filters "Name=tag:CostCenter,Values=123" \
  --query "Subnets[*].SubnetId" \
  --output text
```

Salida:

```
subnet-0987a87c8b37348ef
subnet-02a95061c45f372ee
subnet-03f720e7de2788d73
```

Para obtener más información, consulte [Trabajo con VPC y subredes](#) en la Guía del usuario de Amazon VPC.

- Para obtener más información sobre la API, consulta [DescribeSubnets](#) la Referencia de AWS CLI comandos.

## JavaScript

### SDK para JavaScript (v3)

#### Note

Hay más información. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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 obtener más información sobre la API, consulta [DescribeSubnets](#) la Referencia AWS SDK for JavaScript de la API.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: en este ejemplo se describe la subred especificada.

```
Get-EC2Subnet -SubnetId subnet-1a2b3c4d
```

#### Salida:

```
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
```

Ejemplo 2: En este ejemplo se describen todas las subredes.

```
Get-EC2Subnet
```

- Para obtener más información sobre la API, consulte la referencia [DescribeSubnets](#) de AWS Tools for PowerShell cmdlets.

## Python

### SDK para Python (Boto3)

#### Note

Hay más información al respecto. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

```
class AutoScaler:
    """
    Encapsulates Amazon EC2 Auto Scaling and EC2 management actions.
    """

    def __init__(
        self,
        resource_prefix,
        inst_type,
        ami_param,
        autoscaling_client,
        ec2_client,
        ssm_client,
        iam_client,
    ):
        """
        :param resource_prefix: The prefix for naming AWS resources that are
        created by this class.
        :param inst_type: The type of EC2 instance to create, such as t3.micro.
```

```

        :param ami_param: The Systems Manager parameter used to look up the AMI
that is
                created.
        :param autoscaling_client: A Boto3 EC2 Auto Scaling client.
        :param ec2_client: A Boto3 EC2 client.
        :param ssm_client: A Boto3 Systems Manager client.
        :param iam_client: A Boto3 IAM client.
        """
        self.inst_type = inst_type
        self.ami_param = ami_param
        self.autoscaling_client = autoscaling_client
        self.ec2_client = ec2_client
        self.ssm_client = ssm_client
        self.iam_client = iam_client
        self.launch_template_name = f"{resource_prefix}-template"
        self.group_name = f"{resource_prefix}-group"
        self.instance_policy_name = f"{resource_prefix}-pol"
        self.instance_role_name = f"{resource_prefix}-role"
        self.instance_profile_name = f"{resource_prefix}-prof"
        self.bad_creds_policy_name = f"{resource_prefix}-bc-pol"
        self.bad_creds_role_name = f"{resource_prefix}-bc-role"
        self.bad_creds_profile_name = f"{resource_prefix}-bc-prof"
        self.key_pair_name = f"{resource_prefix}-key-pair"

    def get_subnets(self, vpc_id, zones):
        """
        Gets the default subnets in a VPC for a specified list of Availability
Zones.

        :param vpc_id: The ID of the VPC to look up.
        :param zones: The list of Availability Zones to look up.
        :return: The list of subnets found.
        """
        try:
            response = self.ec2_client.describe_subnets(
                Filters=[
                    {"Name": "vpc-id", "Values": [vpc_id]},
                    {"Name": "availability-zone", "Values": zones},
                    {"Name": "default-for-az", "Values": ["true"]},
                ]
            )
            subnets = response["Subnets"]
            log.info("Found %s subnets for the specified zones.", len(subnets))

```



```
except ClientError as err:
    raise AutoScalerError(f"Couldn't get subnets: {err}")
else:
    return subnets
```

- Para obtener más información sobre la API, consulta [DescribeSubnets](#) la AWS Referencia de API de SDK for Python (Boto3).

Para obtener una lista completa de las guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **DescribeTags** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar DescribeTags.

### CLI

#### AWS CLI

Ejemplo 1: Para describir todas las etiquetas de un único recurso

En el siguiente describe-tags ejemplo, se describen las etiquetas de la instancia especificada.

```
aws ec2 describe-tags \
  --filters "Name=resource-id,Values=i-1234567890abcdef8"
```

Salida:

```
{
  "Tags": [
    {
      "ResourceType": "instance",
      "ResourceId": "i-1234567890abcdef8",
      "Value": "Test",
      "Key": "Stack"
    },
    {
```

```

        "ResourceType": "instance",
        "ResourceId": "i-1234567890abcdef8",
        "Value": "Beta Server",
        "Key": "Name"
    }
]
}

```

**Ejemplo 2:** Para describir todas las etiquetas de un tipo de recurso

En el siguiente `describe-tags` ejemplo, se describen las etiquetas de los volúmenes.

```

aws ec2 describe-tags \
  --filters "Name=resource-type,Values=volume"

```

Salida:

```

{
  "Tags": [
    {
      "ResourceType": "volume",
      "ResourceId": "vol-1234567890abcdef0",
      "Value": "Project1",
      "Key": "Purpose"
    },
    {
      "ResourceType": "volume",
      "ResourceId": "vol-049df61146c4d7901",
      "Value": "Logs",
      "Key": "Purpose"
    }
  ]
}

```

**Ejemplo 3:** Para describir todas las etiquetas

En el siguiente `describe-tags` ejemplo, se describen las etiquetas de todos los recursos.

```

aws ec2 describe-tags

```

**Ejemplo 4:** Para describir las etiquetas de los recursos en función de una clave de etiqueta

En el siguiente `describe-tags` ejemplo, se describen las etiquetas de los recursos que tienen una etiqueta con la clave `Stack`.

```
aws ec2 describe-tags \  
  --filters Name=key,Values=Stack
```

Salida:

```
{  
  "Tags": [  
    {  
      "ResourceType": "volume",  
      "ResourceId": "vol-027552a73f021f3b",  
      "Value": "Production",  
      "Key": "Stack"  
    },  
    {  
      "ResourceType": "instance",  
      "ResourceId": "i-1234567890abcdef8",  
      "Value": "Test",  
      "Key": "Stack"  
    }  
  ]  
}
```

Ejemplo 5: Para describir las etiquetas de sus recursos en función de una clave de etiqueta y un valor de etiqueta

En el siguiente `describe-tags` ejemplo, se describen las etiquetas de los recursos que tienen la etiqueta `Stack=Test`.

```
aws ec2 describe-tags \  
  --filters Name=key,Values=Stack Name=value,Values=Test
```

Salida:

```
{  
  "Tags": [  
    {  
      "ResourceType": "image",
```

```

        "ResourceId": "ami-3ac336533f021f3bd",
        "Value": "Test",
        "Key": "Stack"
    },
    {
        "ResourceType": "instance",
        "ResourceId": "i-1234567890abcdef8",
        "Value": "Test",
        "Key": "Stack"
    }
]
}

```

En el siguiente `describe-tags` ejemplo, se utiliza una sintaxis alternativa para describir los recursos con la etiqueta `Stack=Test`.

```

aws ec2 describe-tags \
  --filters "Name=tag:Stack,Values=Test"

```

En el siguiente `describe-tags` ejemplo, se describen las etiquetas de todas las instancias que tienen una etiqueta con la clave `Purpose` y sin ningún valor.

```

aws ec2 describe-tags \
  --filters "Name=resource-type,Values=instance" "Name=key,Values=Purpose"
  "Name=value,Values="

```

Salida:

```

{
  "Tags": [
    {
      "ResourceType": "instance",
      "ResourceId": "i-1234567890abcdef5",
      "Value": null,
      "Key": "Purpose"
    }
  ]
}

```

- Para obtener información sobre la API, consulta [DescribeTags](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: En este ejemplo se obtienen las etiquetas del tipo de recurso «imagen»

```
Get-EC2Tag -Filter @{Name="resource-type";Values="image"}
```

Salida:

Key	ResourceId	ResourceType	Value
---	-----	-----	----
Name	ami-0a123b4ccb567a8ea	image	Win7-Imported
auto-delete	ami-0a123b4ccb567a8ea	image	never

Ejemplo 2: Este ejemplo busca todas las etiquetas de todos los recursos y las agrupa por tipo de recurso

```
Get-EC2Tag | Group-Object resourcetype
```

Salida:

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}

```

Ejemplo 3: Este ejemplo muestra todos los recursos con la etiqueta 'eliminación automática' con el valor 'no' para la región dada

```
Get-EC2Tag -Region eu-west-1 -Filter @{"Name="tag:auto-delete";Values="no"}
```

Salida:

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

Ejemplo 4: Este ejemplo obtiene todos los recursos con la etiqueta «eliminación automática» con el valor «no» y más filtros en el siguiente tubo para analizar solo los tipos de recursos de «instancia» y, finalmente, crea la etiqueta «ThisInstance» para cada recurso de la instancia, cuyo valor es el propio identificador de la instancia

```
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}}
```

Ejemplo 5: Este ejemplo busca las etiquetas de todos los recursos de la instancia, así como las teclas de «Nombre», y las muestra en formato de tabla

```
Get-EC2Tag -Filter @{"Name="resource-
type";Values="instance"},@{"Name="key";Values="Name"} | Select-Object ResourceId,
@{"Name="Name-Tag";Expression={$PSItem.Value}} | Format-Table -AutoSize
```

Salida:

ResourceId	Name-Tag
-----	-----
i-012e3cb4df567e1aa	jump1
i-01c23a45d6fc7a89f	repro-3

- Para obtener más información sobre la API, consulta la referencia [DescribeTags](#) de AWS Tools for PowerShell cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **DescribeVolumeAttribute** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `DescribeVolumeAttribute`.

### CLI

#### AWS CLI

Para describir un atributo de volumen

Este comando de ejemplo describe el `autoEnableIo` atributo del volumen con el ID `vol-049df61146c4d7901`.

Comando:

```
aws ec2 describe-volume-attribute --volume-id vol-049df61146c4d7901 --attribute autoEnableIO
```

Salida:

```
{
  "AutoEnableIO": {
    "Value": false
  },
  "VolumeId": "vol-049df61146c4d7901"
}
```

- Para obtener más información sobre la API, consulte [DescribeVolumeAttribute](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: en este ejemplo se describe el atributo especificado del volumen especificado.

```
Get-EC2VolumeAttribute -VolumeId vol-12345678 -Attribute AutoEnableIO
```

Salida:

```
AutoEnableIO    ProductCodes    VolumeId
-----
False           {}              vol-12345678
```

- Para obtener más información sobre la API, consulte [DescribeVolumeAttribute](#) la referencia del AWS Tools for PowerShell cmdlet.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **DescribeVolumeStatus** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `DescribeVolumeStatus`.

### CLI

#### AWS CLI

Para describir el estado de un solo volumen

Este comando de ejemplo describe el estado del volumen `vol-1234567890abcdef0`.

Comando:

```
aws ec2 describe-volume-status --volume-ids vol-1234567890abcdef0
```

Salida:

```
{
  "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 describir el estado de los volúmenes deteriorados

Este comando de ejemplo describe el estado de todos los volúmenes deteriorados. En este ejemplo de salida, no hay volúmenes deteriorados.

Comando:

```
aws ec2 describe-volume-status --filters Name=volume-
status.status,Values=impaired
```

Salida:

```
{
  "VolumeStatuses": []
}
```

Si tiene un volumen cuya comprobación de estado ha fallado (el estado es defectuoso), consulte Trabajo con un volumen dañado en la Guía del usuario de Amazon EC2.

- Para obtener más información sobre la API, consulte [DescribeVolumeStatus](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: En este ejemplo se describe el estado del volumen especificado.

```
Get-EC2VolumeStatus -VolumeId vol-12345678
```

Salida:

```
Actions           : {}
AvailabilityZone  : us-west-2a
Events           : {}
VolumeId         : vol-12345678
VolumeStatus     : Amazon.EC2.Model.VolumeStatusInfo
```

```
(Get-EC2VolumeStatus -VolumeId vol-12345678).VolumeStatus
```

Salida:

Details	Status
-----	-----
{io-enabled, io-performance}	ok

```
(Get-EC2VolumeStatus -VolumeId vol-12345678).VolumeStatus.Details
```

Salida:

Name	Status
----	-----
io-enabled	passed
io-performance	not-applicable

- Para obtener más información sobre la API, consulte [DescribeVolumeStatus](#) la referencia del AWS Tools for PowerShell cmdlet.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **DescribeVolumes** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar DescribeVolumes.

### CLI

#### AWS CLI

Ejemplo 1: Para describir un volumen

El siguiente describe-volumes ejemplo describe los volúmenes especificados en la región actual.

```
aws ec2 describe-volumes \  
  --volume-ids vol-049df61146c4d7901 vol-1234567890abcdef0
```

Salida:

```
{  
  "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  
    }  
  ]  
}
```

Ejemplo 2: Para describir los volúmenes que están adjuntos a una instancia específica

En el siguiente `describe-volumes` ejemplo, se describen todos los volúmenes que están adjuntos a la instancia especificada y configurados para que se eliminen cuando la instancia finalice.

```
aws ec2 describe-volumes \  
  --region us-east-1 \  
  --filters Name=attachment.instance-id,Values=i-1234567890abcdef0  
  Name=attachment.delete-on-termination,Values=true
```

Para ver un ejemplo del resultado de `describe-volumes`, consulte el ejemplo 1.

Ejemplo 3: Para describir los volúmenes disponibles en una zona de disponibilidad específica

El siguiente `describe-volumes` ejemplo describe todos los volúmenes que tienen un estado igual o se encuentran en la zona de disponibilidad especificada. `available`

```
aws ec2 describe-volumes \  
  --filters Name=status,Values=available Name=availability-zone,Values=us-  
east-1a
```

Para ver un ejemplo del resultado de `describe-volumes`, consulte el ejemplo 1.

Ejemplo 4: Para describir los volúmenes en función de las etiquetas

El siguiente describe-volumes ejemplo describe todos los volúmenes que tienen la clave de etiqueta Name y un valor que comienza por Test. A continuación, el resultado se filtra con una consulta que muestra únicamente las etiquetas y los ID de los volúmenes.

```
aws ec2 describe-volumes \  
  --filters Name=tag:Name,Values=Test* \  
  --query "Volumes[*].{ID:VolumeId,Tag:Tags}"
```

Salida:

```
[  
  {  
    "Tag": [  
      {  
        "Value": "Test2",  
        "Key": "Name"  
      }  
    ],  
    "ID": "vol-1234567890abcdef0"  
  },  
  {  
    "Tag": [  
      {  
        "Value": "Test1",  
        "Key": "Name"  
      }  
    ],  
    "ID": "vol-049df61146c4d7901"  
  }  
]
```

Para ver ejemplos adicionales de uso de filtros de etiquetas, consulte [Trabajo con etiquetas](#) en la Guía del usuario de Amazon EC2.

- Para obtener más información sobre la API, consulte [DescribeVolumes](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: En este ejemplo se describe el volumen de EBS especificado.

```
Get-EC2Volume -VolumeId vol-12345678
```

**Salida:**

```
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
```

Ejemplo 2: en este ejemplo se describen los volúmenes de EBS que tienen el estado «disponible».

```
Get-EC2Volume -Filter @{ Name="status"; Values="available" }
```

**Salida:**

```
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
...
```

Ejemplo 3: en este ejemplo se describen todos los volúmenes de EBS.

```
Get-EC2Volume
```

- Para obtener más información sobre la API, consulte la referencia [DescribeVolumes](#) de AWS Tools for PowerShell cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **DescribeVpcAttribute** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `DescribeVpcAttribute`.

### CLI

#### AWS CLI

Para describir el `enableDnsSupport` atributo

En este ejemplo se describe el `enableDnsSupport` atributo. Este atributo indica si la resolución de DNS está habilitada para la VPC. Si este atributo es `true`, el servidor DNS de Amazon resuelve nombres de host de DNS para sus instancias en sus direcciones IP correspondientes; de lo contrario, no.

Comando:

```
aws ec2 describe-vpc-attribute --vpc-id vpc-a01106c2 --attribute enableDnsSupport
```

Salida:

```
{
  "VpcId": "vpc-a01106c2",
  "EnableDnsSupport": {
    "Value": true
  }
}
```

Para describir el atributo `enableDnsHostnames`

En este ejemplo se describe el `enableDnsHostnames` atributo. Este atributo indica si las instancias lanzadas en la VPC obtienen nombres de host DNS. Si el atributo es `true`, las instancias de la VPC obtienen nombres de host de DNS; de lo contrario, no.

**Comando:**

```
aws ec2 describe-vpc-attribute --vpc-id vpc-a01106c2 --attribute
enableDnsHostnames
```

**Salida:**

```
{
  "VpcId": "vpc-a01106c2",
  "EnableDnsHostnames": {
    "Value": true
  }
}
```

- Para obtener información sobre la API, consulte la Referencia [DescribeVpcAttribute](#) de AWS CLI comandos.

**PowerShell****Herramientas para PowerShell**

Ejemplo 1: En este ejemplo se describe el atributo enableDnsSupport ".

```
Get-EC2VpcAttribute -VpcId vpc-12345678 -Attribute enableDnsSupport
```

**Salida:**

```
EnableDnsSupport
-----
True
```

Ejemplo 2: En este ejemplo se describe el atributo enableDnsHostnames «».

```
Get-EC2VpcAttribute -VpcId vpc-12345678 -Attribute enableDnsHostnames
```

**Salida:**

```
EnableDnsHostnames
```



```
-----  
True
```

- Para obtener más información sobre la API, consulte [DescribeVpcAttribute](#) la referencia del AWS Tools for PowerShell cmdlet.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **DescribeVpcClassicLink** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar DescribeVpcClassicLink.

### CLI

#### AWS CLI

Para describir el ClassicLink estado de sus VPC

En este ejemplo se muestra el ClassicLink estado de vpc-88888888.

Comando:

```
aws ec2 describe-vpc-classic-link --vpc-id vpc-88888888
```

Salida:

```
{  
  "Vpcs": [  
    {  
      "ClassicLinkEnabled": true,  
      "VpcId": "vpc-88888888",  
      "Tags": [  
        {  
          "Value": "classiclinkvpc",  
          "Key": "Name"  
        }  
      ]  
    }  
  ]  
}
```

```
}

```

En este ejemplo, se muestran solo las VPC que están habilitadas para Classiclink (el valor del `is-classic-link-enabled` filtro está establecido en). `true`

Comando:

```
aws ec2 describe-vpc-classic-link --filter "Name=is-classic-link-
enabled,Values=true"
```

- Para obtener más información sobre la API, consulte la Referencia de [DescribeVpcClassicLink](#) comandos AWS CLI .

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: El ejemplo anterior devuelve todas las VPC de la región con su `ClassicLinkEnabled` estado

```
Get-EC2VpcClassicLink -Region eu-west-1
```

Salida:

```
ClassicLinkEnabled Tags    VpcId
-----
False              {Name} vpc-0fc1ff23f45b678eb
False              {}      vpc-01e23c4a5d6db78e9
False              {Name} vpc-0123456b078b9d01f
False              {}      vpc-12cf3b4f
False              {Name} vpc-0b12d3456a7e8901d
```

- Para obtener más información sobre la API, consulte la referencia [DescribeVpcClassicLink](#) de AWS Tools for PowerShell cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo `DescribeVpcClassicLinkDnsSupport` con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `DescribeVpcClassicLinkDnsSupport`.

### CLI

#### AWS CLI

Para describir el soporte de ClassicLink DNS para sus VPC

En este ejemplo, se describe el estado de soporte de ClassicLink DNS de todas sus VPC.

Comando:

```
aws ec2 describe-vpc-classic-link-dns-support
```

Salida:

```
{
  "Vpcs": [
    {
      "VpcId": "vpc-88888888",
      "ClassicLinkDnsSupported": true
    },
    {
      "VpcId": "vpc-1a2b3c4d",
      "ClassicLinkDnsSupported": false
    }
  ]
}
```

- Para obtener más información sobre la API, consulte [DescribeVpcClassicLinkDnsSupport](#) la Referencia de AWS CLI comandos.

### PowerShell

#### Herramientas para PowerShell

Ejemplo 1: En este ejemplo se describe el estado de soporte de ClassicLink DNS de las VPC de la región eu-west-1

```
Get-EC2VpcClassicLinkDnsSupport -VpcId vpc-0b12d3456a7e8910d -Region eu-west-1
```

Salida:

```
ClassicLinkDnsSupported VpcId
-----
False                   vpc-0b12d3456a7e8910d
False                   vpc-12cf3b4f
```

- Para obtener más información sobre la API, consulte la referencia de [DescribeVpcClassicLinkDnsSupport](#) cmdlets AWS Tools for PowerShell .

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **DescribeVpcEndpointServices** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar DescribeVpcEndpointServices.

CLI

AWS CLI

Ejemplo 1: Para describir todos los servicios de punto final de la VPC

En el siguiente ejemplo «describe-vpc-endpoint-services» se enumeran todos los servicios de punto final de VPC de una AWS región.

```
aws ec2 describe-vpc-endpoint-services
```

Salida:

```
{
  "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 obtener más información, consulte [Ver los nombres de los AWS servicios disponibles](#) en la Guía del usuario de AWS PrivateLink.

**Ejemplo 2:** Para describir los detalles de un servicio de punto final

En el siguiente ejemplo `describe-vpc-endpoint-services` «" se enumeran los detalles del servicio de punto final de la interfaz Amazon S3.

```

aws ec2 describe-vpc-endpoint-services \
  --filter "Name=service-type,Values=Interface" Name=service-
name,Values=com.amazonaws.us-east-1.s3

```

**Salida:**

```
{
  "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 obtener más información, consulte [Ver los nombres de los AWS servicios disponibles](#) en la Guía del usuario de AWS PrivateLink.

- Para obtener más información sobre la API, consulte [DescribeVpcEndpointServices](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: este ejemplo describe el servicio de punto final de VPC de EC2 con el filtro indicado, en este caso `com.amazonaws.eu-west-1.ecs`. Además, también amplía la propiedad y muestra los detalles `ServiceDetails`

```
Get-EC2VpcEndpointService -Region eu-west-1 -MaxResult 5 -Filter @{"Name"="service-name";Values="com.amazonaws.eu-west-1.ecs"} | Select-Object -ExpandProperty ServiceDetails
```

Salida:

```
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
```

Ejemplo 2: Este ejemplo recupera todos los servicios de punto final de la VPC de EC2 y devuelve el `ServiceNames` «`ssm`» coincidente

```
Get-EC2VpcEndpointService -Region eu-west-1 | Select-Object -ExpandProperty Servicenames | Where-Object { -match "ssm"}
```

Salida:

```
com.amazonaws.eu-west-1.ssm
com.amazonaws.eu-west-1.ssmmessages
```

- Para obtener más información sobre la API, consulte [DescribeVpcEndpointServices](#) la referencia de cmdlets.AWS Tools for PowerShell

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.





```
"VpcId": "vpc-1a2b3c4d",
"NetworkInterfaceIds": [
  "eni-2ec2b084",
  "eni-1b4a65cf"
],
"SubnetIds": [
  "subnet-d6fcaa8d",
  "subnet-7b16de0c"
],
"PrivateDnsEnabled": false,
"State": "available",
"ServiceName": "com.amazonaws.us-east-1.elasticloadbalancing",
"RouteTableIds": [],
"Groups": [
  {
    "GroupName": "default",
    "GroupId": "sg-54e8bf31"
  }
],
"VpcEndpointId": "vpce-0f89a33420c1931d7",
"VpcEndpointType": "Interface",
"CreationTimestamp": "2017-09-05T17:55:27.583Z",
"DnsEntries": [
  {
    "HostedZoneId": "Z7HUB22UULQXV",
    "DnsName": "vpce-0f89a33420c1931d7-
bluzidnv.elasticloadbalancing.us-east-1.vpce.amazonaws.com"
  },
  {
    "HostedZoneId": "Z7HUB22UULQXV",
    "DnsName": "vpce-0f89a33420c1931d7-bluzidnv-us-
east-1b.elasticloadbalancing.us-east-1.vpce.amazonaws.com"
  },
  {
    "HostedZoneId": "Z7HUB22UULQXV",
    "DnsName": "vpce-0f89a33420c1931d7-bluzidnv-us-
east-1a.elasticloadbalancing.us-east-1.vpce.amazonaws.com"
  }
],
"OwnerId": "123456789012"
},
{
  "VpcEndpointId": "vpce-aabbaabbaabbaabba",
  "VpcEndpointType": "GatewayLoadBalancer",
```

```
    "VpcId": "vpc-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 obtener más información, consulte [Puntos de enlace de la VPC](#) en la Guía del usuario de Amazon VPC.

- Para obtener información sobre la API, consulte la Referencia [DescribeVpcEndpoints](#) de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: En este ejemplo se describen uno o varios de sus puntos de enlace de VPC para la región eu-west-1. A continuación, canaliza el resultado al siguiente comando, que selecciona la `VpcEndpointId` propiedad y devuelve el ID de VPC de la matriz como matriz de cadenas.

```
Get-EC2VpcEndpoint -Region eu-west-1 | Select-Object -ExpandProperty
VpcEndpointId
```

### Salida:

```
vpce-01a2ab3f4f5cc6f7d
vpce-01d2b345a6787890b
vpce-0012e34d567890e12
vpce-0c123db4567890123
```

Ejemplo 2: Este ejemplo describe todos los puntos finales de vpc de la región eu-west-1 y selecciona VpcEndpointId VpcId, ServiceName y PrivateDnsEnabled propiedades para presentarlos en formato tabular

```
Get-EC2VpcEndpoint -Region eu-west-1 | Select-Object VpcEndpointId, VpcId,
  ServiceName, PrivateDnsEnabled | Format-Table -AutoSize
```

Salida:

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.ssmmessages
PrivateDnsEnabled		
True		

Ejemplo 3: Este ejemplo exporta el documento de política del punto final de VPC vpce-01a2ab3f4f5cc6f7d a un archivo json

```
Get-EC2VpcEndpoint -Region eu-west-1 -VpcEndpointId vpce-01a2ab3f4f5cc6f7d |
  Select-Object -expand PolicyDocument | Out-File vpce_policyDocument.json
```

- Para obtener [DescribeVpcEndpoints](#) más AWS Tools for PowerShell información sobre la API, consulte la referencia de cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **DescribeVpcs** con un AWS SDK o CLI


En los siguientes ejemplos de código, se muestra cómo utilizar DescribeVpcs.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en contexto en el siguiente ejemplo de código:

- [Creación y administración de un servicio resiliente](#)

.NET

AWS SDK for .NET

 Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

```
/// <summary>
/// Get the default VPC for the account.
/// </summary>
/// <returns>The default VPC object.</returns>
public async Task<Vpc> GetDefaultVpc()
{
    var vpcResponse = await _amazonEc2.DescribeVpcsAsync(
        new DescribeVpcsRequest()
        {
            Filters = new List<Amazon.EC2.Model.Filter>()
            {
                new ("is-default", new List<string>() { "true" })
            }
        });
    return vpcResponse.Vpcs[0];
}
```

- Para obtener más información sobre la API, consulta [DescribeVpcs](#) la Referencia AWS SDK for .NET de la API.

## CLI

## AWS CLI

## Ejemplo 1: Describir todas las VPC

En el siguiente ejemplo de `describe-vpcs`, se recuperan los detalles de todas las VPC.

```
aws ec2 describe-vpcs
```

Salida:

```
{
  "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"
      }
    ]
  }
]
}

```

## Ejemplo 2: Describir una VPC específica

En siguiente ejemplo, `describe-vpcs` recupera los detalles de la VPC especificada.

```

aws ec2 describe-vpcs \
  --vpc-ids vpc-06e4ab6c6cEXAMPLE

```

Salida:

```

{
  "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 obtener más información sobre la API, consulta [DescribeVpcs](#) la Referencia de AWS CLI comandos.

## JavaScript

### SDK para JavaScript (v3)

#### Note

Hay más información. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

```

const client = new EC2Client({});
const { Vpcs } = await client.send(
    new DescribeVpcsCommand({
        Filters: [{ Name: "is-default", Values: ["true"] }]},
    ),
);

```

- Para obtener más información sobre la API, consulta [DescribeVpcs](#) la Referencia AWS SDK for JavaScript de la API.



## PowerShell

### Herramientas para PowerShell

Ejemplo 1: En este ejemplo se describe la VPC especificada.

```
Get-EC2Vpc -VpcId vpc-12345678
```

Salida:

```
CidrBlock      : 10.0.0.0/16
DhcpOptionsId  : dopt-1a2b3c4d
InstanceTenancy : default
IsDefault      : False
State          : available
Tags           : {Name}
VpcId          : vpc-12345678
```

Ejemplo 2: en este ejemplo se describe la VPC predeterminada (solo puede haber una por región). Si su cuenta es compatible con EC2-Classic en esta región, no hay una VPC predeterminada.

```
Get-EC2Vpc -Filter @{{Name="isDefault"; Values="true"}}
```

Salida:

```
CidrBlock      : 172.31.0.0/16
DhcpOptionsId  : dopt-12345678
InstanceTenancy : default
IsDefault      : True
State          : available
Tags           : {}
VpcId          : vpc-45678901
```

Ejemplo 3: En este ejemplo se describen las VPC que coinciden con el filtro especificado (es decir, tienen un CIDR que coincide con el valor «10.0.0.0/16» y están en el estado «disponibles»).

```
Get-EC2Vpc -Filter @{{Name="cidr";
  Values="10.0.0.0/16"},@{{Name="state";Values="available"}}
```

Ejemplo 4: En este ejemplo se describen todas las VPC.

Get-EC2Vpc

- Para obtener información sobre la API, consulte la referencia [DescribeVpcs](#) de AWS Tools for PowerShell cmdlets.

## Python

### SDK para Python (Boto3)

#### Note

Hay más información al respecto. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

```
class AutoScaler:
    """
    Encapsulates Amazon EC2 Auto Scaling and EC2 management actions.
    """

    def __init__(
        self,
        resource_prefix,
        inst_type,
        ami_param,
        autoscaling_client,
        ec2_client,
        ssm_client,
        iam_client,
    ):
        """
        :param resource_prefix: The prefix for naming AWS resources that are
        created by this class.
        :param inst_type: The type of EC2 instance to create, such as t3.micro.
        :param ami_param: The Systems Manager parameter used to look up the AMI
        that is
                created.
        :param autoscaling_client: A Boto3 EC2 Auto Scaling client.
        :param ec2_client: A Boto3 EC2 client.
```

```
:param ssm_client: A Boto3 Systems Manager client.
:param iam_client: A Boto3 IAM client.
"""
self.inst_type = inst_type
self.ami_param = ami_param
self.autoscaling_client = autoscaling_client
self.ec2_client = ec2_client
self.ssm_client = ssm_client
self.iam_client = iam_client
self.launch_template_name = f"{resource_prefix}-template"
self.group_name = f"{resource_prefix}-group"
self.instance_policy_name = f"{resource_prefix}-pol"
self.instance_role_name = f"{resource_prefix}-role"
self.instance_profile_name = f"{resource_prefix}-prof"
self.bad_creds_policy_name = f"{resource_prefix}-bc-pol"
self.bad_creds_role_name = f"{resource_prefix}-bc-role"
self.bad_creds_profile_name = f"{resource_prefix}-bc-prof"
self.key_pair_name = f"{resource_prefix}-key-pair"

def get_default_vpc(self):
    """
    Gets the default VPC for the account.

    :return: Data about the default VPC.
    """
    try:
        response = self.ec2_client.describe_vpcs(
            Filters=[{"Name": "is-default", "Values": ["true"]}])
    except ClientError as err:
        raise AutoScalerError(f"Couldn't get default VPC: {err}")
    else:
        return response["Vpcs"][0]
```

- Para obtener más información sobre la API, consulta [DescribeVpcs](#) la AWS Referencia de API de SDK for Python (Boto3).

Para obtener una lista completa de las guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **DescribeVpnConnections** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar DescribeVpnConnections.

### CLI

#### AWS CLI

Ejemplo 1: Para describir sus conexiones VPN

En el siguiente describe-vpn-connections ejemplo, se describen todas las conexiones de VPN de Site-to-Site.

```
aws ec2 describe-vpn-connections
```

Salida:

```
{
  "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 obtener más información, consulte [Cómo funciona la VPN de AWS sitio a sitio en la Guía del usuario de VPN](#) de sitio a sitio.AWS

Ejemplo 2: Para describir las conexiones VPN disponibles

En el siguiente `describe-vpn-connections` ejemplo, se describen las conexiones VPN de Site-to-Site con un estado de `available`

```

aws ec2 describe-vpn-connections \
  --filters "Name=state,Values=available"

```

Para obtener más información, consulte [Cómo funciona la VPN de AWS sitio a sitio en la Guía del usuario de VPN](#) de sitio a sitio.AWS

- Para obtener más información sobre la API, consulte [DescribeVpnConnections](#) la Referencia de comandos.AWS CLI

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: En este ejemplo se describe la conexión VPN especificada.

```
Get-EC2VpnConnection -VpnConnectionId vpn-12345678
```

Salida:

```
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
```

Ejemplo 2: Este ejemplo describe cualquier conexión VPN cuyo estado esté pendiente o disponible.

```
$filter = New-Object Amazon.EC2.Model.Filter
$filter.Name = "state"
$filter.Values = @( "pending", "available" )

Get-EC2VpnConnection -Filter $filter
```

Ejemplo 3: En este ejemplo se describen todas sus conexiones VPN.

```
Get-EC2VpnConnection
```

- Para obtener más información sobre la API, consulte [DescribeVpnConnections](#) la referencia de AWS Tools for PowerShell cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **DescribeVpnGateways** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `DescribeVpnGateways`.

### CLI

#### AWS CLI

Para describir sus puertas de enlace privadas virtuales

En este ejemplo se describen sus puertas de enlace privadas virtuales.

Comando:

```
aws ec2 describe-vpn-gateways
```

Salida:

```
{
  "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 obtener más información sobre la API, consulte [DescribeVpnGateways](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: en este ejemplo se describe la puerta de enlace privada virtual especificada.

```
Get-EC2VpnGateway -VpnGatewayId vgw-1a2b3c4d
```

Salida:

```

AvailabilityZone :
State           : available
Tags            : {}
Type            : ipsec.1
VpcAttachments  : {vpc-12345678}
VpnGatewayId    : vgw-1a2b3c4d

```

Ejemplo 2: Este ejemplo describe cualquier puerta de enlace privada virtual cuyo estado esté pendiente o disponible.

```

$filter = New-Object Amazon.EC2.Model.Filter
$filter.Name = "state"
$filter.Values = @( "pending", "available" )

Get-EC2VpnGateway -Filter $filter

```

Ejemplo 3: En este ejemplo se describen todas las puertas de enlace privadas virtuales.

```
Get-EC2VpnGateway
```

- Para obtener información sobre la API, consulte la referencia [DescribeVpnGateways](#) de AWS Tools for PowerShell cmdlets.



Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte [Cree recursos de Amazon EC2 mediante un SDK AWS](#). En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo `DetachInternetGateway` con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `DetachInternetGateway`.

### CLI

#### AWS CLI

Para separar una puerta de enlace a Internet de la VPC

En el siguiente `detach-internet-gateway` ejemplo, se separa la puerta de enlace de Internet especificada de la VPC específica.

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

Este comando no genera ninguna salida.

Para obtener más información, consulte [Puertas de enlace de Internet](#) en la Guía del usuario de Amazon VPC.

- Para obtener información sobre la API, consulte la Referencia [DetachInternetGateway](#) de AWS CLI comandos.

### PowerShell

#### Herramientas para PowerShell

Ejemplo 1: en este ejemplo se separa la puerta de enlace de Internet especificada de la VPC especificada.

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

- Para obtener más información sobre la API, consulte la referencia de [DetachInternetGateway AWS Tools for PowerShell](#) cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **DetachNetworkInterface** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar DetachNetworkInterface.

### CLI

#### AWS CLI

Para separar una interfaz de red de la instancia

En este ejemplo, se separa la interfaz de red especificada de la instancia especificada. Si el comando se ejecuta correctamente, no se muestra ningún resultado.

Comando:

```
aws ec2 detach-network-interface --attachment-id eni-attach-66c4350a
```

- Para obtener más información sobre la API, consulte [DetachNetworkInterface](#) la Referencia de AWS CLI comandos.

### PowerShell

#### Herramientas para PowerShell

Ejemplo 1: en este ejemplo se elimina el adjunto especificado entre una interfaz de red y una instancia.

```
Dismount-EC2NetworkInterface -AttachmentId eni-attach-1a2b3c4d -Force
```

- Para obtener más información sobre la API, consulte [DetachNetworkInterface](#) la referencia de AWS Tools for PowerShell cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **DetachVolume** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar DetachVolume.

### CLI

#### AWS CLI

Para separar un volumen de una instancia

Este comando de ejemplo separa el volumen (vol-049df61146c4d7901) de la instancia a la que está conectado.

Comando:

```
aws ec2 detach-volume --volume-id vol-1234567890abcdef0
```

Salida:

```
{
  "AttachTime": "2014-02-27T19:23:06.000Z",
  "InstanceId": "i-1234567890abcdef0",
  "VolumeId": "vol-049df61146c4d7901",
  "State": "detaching",
  "Device": "/dev/sdb"
}
```

- Para obtener más información sobre la API, consulta [DetachVolume](#) la Referencia de AWS CLI comandos.

### PowerShell

#### Herramientas para PowerShell

Ejemplo 1: En este ejemplo se separa el volumen especificado.

```
Dismount-EC2Volume -VolumeId vol-12345678
```

Salida:

```
AttachTime      : 12/22/2015 1:53:58 AM
DeleteOnTermination : False
Device          : /dev/sdh
InstanceId      : i-1a2b3c4d
State           : detaching
VolumeId       : vol-12345678
```

Ejemplo 2: También puede especificar el ID de la instancia y el nombre del dispositivo para asegurarse de que está separando el volumen correcto.

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

- Para obtener más información sobre la API, consulte la referencia [DetachVolume](#) del AWS Tools for PowerShell cmdlet.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **DetachVpnGateway** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar DetachVpnGateway.

### CLI

#### AWS CLI

Para separar una puerta de enlace privada virtual de su VPC

En este ejemplo, se separa la puerta de enlace privada virtual especificada de la VPC especificada. Si el comando se ejecuta correctamente, no se muestra ningún resultado.

Comando:

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

- Para obtener información sobre la API, consulte la Referencia [DetachVpnGateway](#) de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: en este ejemplo se separa la puerta de enlace privada virtual especificada de la VPC especificada.

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

- Para obtener más información sobre la API, consulte la referencia de [DetachVpnGateway](#) [AWS Tools for PowerShell](#) cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **DisableVgwRoutePropagation** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `DisableVgwRoutePropagation`.

### CLI

#### AWS CLI

Para deshabilitar la propagación de rutas

Este ejemplo impide que la puerta de enlace privada virtual especificada propague rutas estáticas a la tabla de rutas especificada. Si el comando se ejecuta correctamente, no se muestra ningún resultado.

Comando:

```
aws ec2 disable-vgw-route-propagation --route-table-id rtb-22574640 --gateway-id vgw-9a4cacf3
```

- Para obtener más información sobre la API, consulte la Referencia [DisableVgwRoutePropagation](#) de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: Este ejemplo impide que el VGW propague automáticamente las rutas a la tabla de enrutamiento especificada.

```
Disable-EC2VgwRoutePropagation -RouteTableId rtb-12345678 -GatewayId vgw-1a2b3c4d
```

- Para obtener más información sobre la API, consulte [DisableVgwRoutePropagation](#) la referencia del cmdlet.AWS Tools for PowerShell

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **DisableVpcClassicLink** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `DisableVpcClassicLink`.

### CLI

#### AWS CLI

ClassicLink Para inhabilitar una VPC

En este ejemplo, se deshabilita la ClassicLink vpc-88888888.

Comando:

```
aws ec2 disable-vpc-classic-link --vpc-id vpc-88888888
```

Salida:

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

- Para obtener más información sobre la API, consulte la Referencia de [DisableVpcClassicLink](#) comandos AWS CLI .

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: Este ejemplo deshabilita EC2 VpcClassicLink para el vpc-01e23c4a5d6db78e9. Devuelve Verdadero o Falso

```
Disable-EC2VpcClassicLink -VpcId vpc-01e23c4a5d6db78e9
```

- Para obtener más información sobre la API, consulte [DisableVpcClassicLink](#) la referencia de AWS Tools for PowerShell cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **DisableVpcClassicLinkDnsSupport** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `DisableVpcClassicLinkDnsSupport`.

### CLI

#### AWS CLI

Para inhabilitar la compatibilidad de ClassicLink DNS para una VPC

En este ejemplo, se deshabilita la compatibilidad con ClassicLink DNS para. vpc-88888888

Comando:

```
aws ec2 disable-vpc-classic-link-dns-support --vpc-id vpc-88888888
```

Salida:

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

- Para obtener más información sobre la API, consulte [DisableVpcClassicLinkDnsSupport](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: Este ejemplo deshabilita la compatibilidad con ClassicLink DNS para el vpc-0b12d3456a7e8910d

```
Disable-EC2VpcClassicLinkDnsSupport -VpcId vpc-0b12d3456a7e8910d
```

- Para obtener más información [DisableVpcClassicLinkDnsSupport](#) sobre AWS Tools for PowerShell la API, consulte la referencia del cmdlet.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **DisassociateAddress** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `DisassociateAddress`.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en contexto en el siguiente ejemplo de código:

- [Comience a utilizar instancias](#)

## .NET

### AWS SDK for .NET

#### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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)
{
    var response = await _amazonEC2.DisassociateAddressAsync(
        new DisassociateAddressRequest { AssociationId = associationId });
    return response.HttpStatusCode == HttpStatusCode.OK;
}
```

- Para obtener más información sobre la API, consulta [DisassociateAddress](#) la Referencia AWS SDK for .NET de la API.

## Bash

### AWS CLI con el script Bash

#### Note

Hay más información. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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
}
```

## Las funciones de utilidad utilizadas en este ejemplo.

```
#####
# 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 obtener más información sobre la API, consulta [DisassociateAddress](#) la Referencia de AWS CLI comandos.

## CLI

### AWS CLI

#### Desasociar una dirección IP elástica en EC2-Classic

En este ejemplo, se desasocia una dirección IP elástica de una instancia en EC2-Classic. Si el comando se ejecuta correctamente, no se muestra ningún resultado.

Comando:

```
aws ec2 disassociate-address --public-ip 198.51.100.0
```

#### Desasociar una dirección IP elástica en EC2-VPC

En este ejemplo, se desasocia una dirección IP elástica de una instancia en una VPC. Si el comando se ejecuta correctamente, no se muestra ningún resultado.

Comando:

```
aws ec2 disassociate-address --association-id eipassoc-2bebb745
```

- Para obtener más información sobre la API, consulte [DisassociateAddress](#) la Referencia de AWS CLI comandos.

## Java

### SDK para Java 2.x

#### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

```
public static void disassociateAddress(Ec2Client ec2, String associationId) {
    try {
        DisassociateAddressRequest addressRequest =
DisassociateAddressRequest.builder()
            .associationId(associationId)
            .build();

        ec2.disassociateAddress(addressRequest);
        System.out.println("You successfully disassociated the address!");

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

- Para obtener más información sobre la API, consulta [DisassociateAddress](#) la Referencia AWS SDK for Java 2.x de la API.

## JavaScript

### SDK para JavaScript (v3)

#### Note

Hay más información. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

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

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

// Disassociate an Elastic IP address from an instance.
export const main = async () => {
    const command = new DisassociateAddressCommand({
        // You can also use PublicIp, but that is for EC2 classic which is being
        retired.
        AssociationId: "ASSOCIATION_ID",
    });
};
```

```
try {
    await client.send(command);
    console.log("Successfully disassociated address");
} catch (err) {
    console.error(err);
}
};
```

- Para obtener más información sobre la API, consulta [DisassociateAddress](#) la Referencia AWS SDK for JavaScript de la API.

## Kotlin

### SDK para Kotlin

#### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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 obtener más información sobre la API, consulta [DisassociateAddress](#) la referencia sobre el AWS SDK para la API de Kotlin.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: este ejemplo desasocia la dirección IP elástica especificada de la instancia especificada en una VPC.

```
Unregister-EC2Address -AssociationId eipassoc-12345678
```

Ejemplo 2: Este ejemplo desasocia la dirección IP elástica especificada de la instancia especificada en EC2-Classic.

```
Unregister-EC2Address -PublicIp 203.0.113.17
```

- Para obtener más información sobre la API, consulte la referencia del [DisassociateAddress](#) cmdlet AWS Tools for PowerShell .

## Python

### SDK para Python (Boto3)

#### Note

Hay más información al respecto. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

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

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

```

        wraps Elastic IP actions.

        """
        self.ec2_resource = ec2_resource
        self.elastic_ip = elastic_ip

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

    def disassociate(self):
        """
        Removes an association between an Elastic IP address and an instance.
        When the
        association is removed, the instance is assigned a new public IP address.
        """
        if self.elastic_ip is None:
            logger.info("No Elastic IP to disassociate.")
            return

        try:
            self.elastic_ip.association.delete()
        except ClientError as err:
            logger.error(
                "Couldn't disassociate Elastic IP %s from its instance. Here's
why: %s: %s",
                self.elastic_ip.allocation_id,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise

```

- Para obtener más información sobre la API, consulta [DisassociateAddress](#) la AWS Referencia de API de SDK for Python (Boto3).

Para obtener una lista completa de las guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.



## Úselo **DisassociateRouteTable** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `DisassociateRouteTable`.

### CLI

#### AWS CLI

Para desasociar una tabla de rutas

Este ejemplo desasocia la tabla de rutas especificada de la subred especificada. Si el comando se ejecuta correctamente, no se muestra ningún resultado.

Comando:

```
aws ec2 disassociate-route-table --association-id rtbassoc-781d0d1a
```

- Para obtener más información sobre la API, consulte la Referencia [DisassociateRouteTable](#) de AWS CLI comandos.

### PowerShell

#### Herramientas para PowerShell

Ejemplo 1: Este ejemplo elimina la asociación especificada entre una tabla de enrutamiento y una subred.

```
Unregister-EC2RouteTable -AssociationId rtbassoc-1a2b3c4d
```

- Para obtener más información sobre la API, consulte la referencia [DisassociateRouteTable](#) de AWS Tools for PowerShell cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **EnableVgwRoutePropagation** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `EnableVgwRoutePropagation`.

## CLI

### AWS CLI

Para habilitar la propagación de rutas

Este ejemplo permite que la puerta de enlace privada virtual especificada propague las rutas estáticas a la tabla de rutas especificada. Si el comando se ejecuta correctamente, no se muestra ningún resultado.

Comando:

```
aws ec2 enable-vgw-route-propagation --route-table-id rtb-22574640 --gateway-id vgw-9a4cacf3
```

- Para obtener más información sobre la API, consulte [EnableVgwRoutePropagation](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: Este ejemplo permite al VGW especificado propagar las rutas automáticamente a la tabla de enrutamiento especificada.

```
Enable-EC2VgwRoutePropagation -RouteTableId rtb-12345678 -GatewayId vgw-1a2b3c4d
```

- Para obtener más información sobre la API, consulte la referencia del [EnableVgwRoutePropagation](#) cmdlet AWS Tools for PowerShell .

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **EnableVolumeIo** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar EnableVolumeIo.

## CLI

### AWS CLI

Para habilitar la E/S de un volumen

En este ejemplo, se habilita la E/S en el volumen. `vol-1234567890abcdef0`

Comando:

```
aws ec2 enable-volume-io --volume-id vol-1234567890abcdef0
```

Salida:

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

- Para obtener más información sobre la API, consulte [EnableVolume](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: Este ejemplo habilita las operaciones de E/S para el volumen especificado, si las operaciones de E/S estaban deshabilitadas.

```
Enable-EC2VolumeIO -VolumeId vol-12345678
```

- Para obtener más información sobre la API, consulte la referencia de [EnableVolume AWS Tools for PowerShell](#) cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **EnableVpcClassicLink** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `EnableVpcClassicLink`.

## CLI

### AWS CLI

Para habilitar una VPC para ClassicLink

Este ejemplo habilita vpc-88888888 para ClassicLink

Comando:

```
aws ec2 enable-vpc-classic-link --vpc-id vpc-88888888
```

Salida:

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

- Para obtener más información sobre la API, consulte la Referencia [EnableVpcClassicLink](#) de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: Este ejemplo habilita la VPC vpc-0123456b789b0d12f para ClassicLink

```
Enable-EC2VpcClassicLink -VpcId vpc-0123456b789b0d12f
```

Salida:

```
True
```

- Para obtener información sobre [EnableVpcClassicLink](#) la AWS Tools for PowerShell API, consulte la referencia del cmdlet.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **EnableVpcClassicLinkDnsSupport** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `EnableVpcClassicLinkDnsSupport`.

### CLI

#### AWS CLI

Para habilitar el soporte de ClassicLink DNS para una VPC

Este ejemplo habilita el soporte de ClassicLink DNS para `vpc-88888888`.

Comando:

```
aws ec2 enable-vpc-classic-link-dns-support --vpc-id vpc-88888888
```

Salida:

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

- Para obtener más información sobre la API, consulte [EnableVpcClassicLinkDnsSupport](#) la Referencia de AWS CLI comandos.

### PowerShell

#### Herramientas para PowerShell

Ejemplo 1: Este ejemplo permite que `vpc-0b12d3456a7e8910d` admita la resolución de nombres de host DNS para ClassicLink

```
Enable-EC2VpcClassicLinkDnsSupport -VpcId vpc-0b12d3456a7e8910d -Region eu-west-1
```

- Para obtener más [EnableVpcClassicLinkDnsSupport](#) información AWS Tools for PowerShell sobre la API, consulte la referencia de cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **GetConsoleOutput** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `GetConsoleOutput`.

### CLI

#### AWS CLI

Ejemplo 1: Para obtener el resultado de la consola

El siguiente `get-console-output` ejemplo obtiene el resultado de la consola para la instancia de Linux especificada.

```
aws ec2 get-console-output \  
  --instance-id i-1234567890abcdef0
```

Salida:

```
{  
  "InstanceId": "i-1234567890abcdef0",  
  "Timestamp": "2013-07-25T21:23:53.000Z",  
  "Output": "..."  
}
```

Para obtener más información, consulte la [salida de la consola de instancias](#) en la Guía del usuario de Amazon EC2.

Ejemplo 2: Para obtener el último resultado de la consola

En el siguiente `get-console-output` ejemplo, se obtiene el último resultado de la consola para la instancia de Linux especificada.

```
aws ec2 get-console-output \  
  --instance-id i-1234567890abcdef0 \  
  --latest \  
  --output text
```

Salida:

```
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 obtener más información, consulte la [salida de la consola de instancias](#) en la Guía del usuario de Amazon EC2.

- Para obtener más información sobre la API, consulte [GetConsoleOutput](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: en este ejemplo se obtiene el resultado de la consola para la instancia de Linux especificada. La salida de la consola está codificada.

```
Get-EC2ConsoleOutput -InstanceId i-0e19abcd47c123456
```

Salida:

InstanceId	Output
-----	-----
i-0e194d3c47c123637	WyAgICAwLjAwMDAwMF0gQ29tbW...bGU9dHR5UzAgc2Vs

Ejemplo 2: Este ejemplo almacena la salida de la consola codificada en una variable y, a continuación, la decodifica.

```
$Output_encoded = (Get-EC2ConsoleOutput -InstanceId i-0e19abcd47c123456).Output
[System.Text.Encoding]::UTF8.GetString([System.Convert]::FromBase64String($Output_encoded))
```

- Para obtener más información sobre la API, consulte la referencia [GetConsoleOutput](#) de AWS Tools for PowerShell cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **GetHostReservationPurchasePreview** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `GetHostReservationPurchasePreview`.

### CLI

#### AWS CLI

Para obtener una vista previa de la compra de una reserva de anfitrión dedicado

En este ejemplo, se proporciona una vista previa de los costes de una reserva de anfitrión dedicado específica para el anfitrión dedicado especificado en su cuenta.

Comando:

```
aws ec2 get-host-reservation-purchase-preview --offering-id hro-03f707bf363b6b324
--host-id-set h-013abcd2a00cbd123
```

Salida:

```
{
  "TotalHourlyPrice": "1.499",
  "Purchase": [
    {
      "HourlyPrice": "1.499",
      "InstanceFamily": "m4",
      "PaymentOption": "NoUpfront",
      "HostIdSet": [
        "h-013abcd2a00cbd123"
      ],
      "UpfrontPrice": "0.000",
      "Duration": 31536000
    }
  ],
  "TotalUpfrontPrice": "0.000"
}
```



- Para obtener más información sobre la API, consulte [GetHostReservationPurchasePreview](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: En este ejemplo se muestra una vista previa de la compra de una reserva con configuraciones que coinciden con las de su servidor dedicado h-01e23f4cd567890f1

```
Get-EC2HostReservationPurchasePreview -OfferingId hro-0c1f23456789d0ab -HostIdSet
h-01e23f4cd567890f1
```

Salida:

```
CurrencyCode Purchase TotalHourlyPrice TotalUpfrontPrice
-----
                {}          1.307          0.000
```

- Para obtener [GetHostReservationPurchasePreview](#) más AWS Tools for PowerShell información sobre la API, consulte la referencia de cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **GetPasswordData** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar GetPasswordData.

### CLI

#### AWS CLI

Para obtener la contraseña cifrada

En este ejemplo se obtiene la contraseña cifrada.

Comando:

```
aws ec2 get-password-data --instance-id i-1234567890abcdef0
```

Salida:

```
{
  "InstanceId": "i-1234567890abcdef0",
  "Timestamp": "2013-08-07T22:18:38.000Z",
  "PasswordData": "gSlJFq+VpcZXqy+iktXMF6NyxQ4qCrT4+ga0uN0enX1MmgXPTj7XEXAMPLE
UQ+YeFfb+L1U4C4AKv652Ux1iRB3CPTYP7WmU3TUnhsuBd+p6LVk7T2lKUm160Xbk6WPW1VYYm/TRPB1
e1DQ7PY4an/DgZT4mwcpRFIGzhniQgDDe01InvSDcwoUTwNs0Y1S8ouri2W4n5GNlriM3Q0AnNVe1Vz/
53TkDtxbNoU606M1gK9zUWSxqEgwvbV2j8c5rP0WCuaMWSF14ziDu4bd7q+4RSyi8NUsVWnKZ4aEZffu
DPGzKrF5yL1f3etP2L4ZR6CvG7K1hx7VK0QVN32Dajw=="
}
```

Para obtener la contraseña descifrada

En este ejemplo se obtiene la contraseña descifrada.

Comando:

```
aws ec2 get-password-data --instance-id i-1234567890abcdef0 --priv-launch-key C:
\Keys\MyKeyPair.pem
```

Salida:

```
{
  "InstanceId": "i-1234567890abcdef0",
  "Timestamp": "2013-08-30T23:18:05.000Z",
  "PasswordData": "&ViJ652e*u"
}
```

- Para obtener más información sobre la API, consulte [GetPasswordData](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: en este ejemplo se descifra la contraseña que Amazon EC2 asignó a la cuenta de administrador de la instancia de Windows especificada. Cuando se especificó un archivo pem, se asume automáticamente la configuración del conmutador `-Decrypt`.

```
Get-EC2PasswordData -InstanceId i-12345678 -PemFile C:\path\my-key-pair.pem
```

Salida:

```
mYZ(PA9?C)Q
```

Ejemplo 2: ( PowerShell solo en Windows) Inspecciona la instancia para determinar el nombre del par de claves utilizado para lanzar la instancia y, a continuación, intenta buscar los datos del par de claves correspondientes en el almacén de configuración del AWS Toolkit for Visual Studio. Si se encuentran los datos del par de claves, se descifra la contraseña.

```
Get-EC2PasswordData -InstanceId i-12345678 -Decrypt
```

Salida:

```
mYZ(PA9?C)Q
```

Ejemplo 3: Devuelve los datos de contraseña cifrados de la instancia.

```
Get-EC2PasswordData -InstanceId i-12345678
```

Salida:

```
iVz3BAK/WAXV.....dqt8WeMA==
```

- Para obtener más información sobre la API, consulte [GetPasswordData](#) la referencia de AWS Tools for PowerShell cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **ImportImage** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `ImportImage`.

## CLI

### AWS CLI

Para importar un archivo de imagen de máquina virtual como AMI

En el siguiente `import-image` ejemplo, se importa la OVA especificada.

```
aws ec2 import-image \  
  --disk-containers Format=ova,UserBucket="{S3Bucket=my-import-bucket,S3Key=vms/  
my-server-vm.ova}"
```

Salida:

```
{  
  "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 obtener más información sobre la API, consulte [ImportImage](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: Este ejemplo importa una imagen de máquina virtual de disco único del bucket de Amazon S3 especificado a Amazon EC2 con un token de idempotencia. El ejemplo requiere que exista un rol de servicio de importación de VM con el nombre predeterminado

«vmimport», con una política que permita a Amazon EC2 acceder al bucket especificado, como se explica en el tema Requisitos previos de VM Import. Para usar un rol personalizado, especifique el nombre del rol mediante el parámetro. **-RoleName**

```
$container = New-Object Amazon.EC2.Model.ImageDiskContainer
$container.Format="VMDK"
$container.UserBucket = New-Object Amazon.EC2.Model.UserBucket
$container.UserBucket.S3Bucket = "myVirtualMachineImages"
$container.UserBucket.S3Key = "Win_2008_Server_Standard_SP2_64-bit-disk1.vmdk"

$params = @{
    "ClientToken"="idempotencyToken"
    "Description"="Windows 2008 Standard Image Import"
    "Platform"="Windows"
    "LicenseType"="AWS"
}

Import-EC2Image -DiskContainer $container @params
```

Salida:

```
Architecture      :
Description       : Windows 2008 Standard Image
Hypervisor        :
ImageId           :
ImportTaskId      : import-ami-abcdefgh
LicenseType       : AWS
Platform          : Windows
Progress          : 2
SnapshotDetails   : {}
Status            : active
StatusMessage     : pending
```

- Para obtener más información sobre la API, consulte [ImportImage](#) la referencia de AWS Tools for PowerShell cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **ImportKeyPair** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `ImportKeyPair`.

### CLI

#### AWS CLI

Para importar una clave pública

Primero, genere un key pair con la herramienta que prefiera. Por ejemplo, utilice este comando `ssh-keygen`:

Comando:

```
ssh-keygen -t rsa -C "my-key" -f ~/.ssh/my-key
```

Salida:

```
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 comando de ejemplo importa la clave pública especificada.

Comando:

```
aws ec2 import-key-pair --key-name "my-key" --public-key-material fileb://~/.ssh/  
my-key.pub
```

Salida:

```
{  
  "KeyName": "my-key",  
  "KeyFingerprint": "1f:51:ae:28:bf:89:e9:d8:1f:25:5d:37:2d:7d:b8:ca"  
}
```

- Para obtener más información sobre la API, consulte [ImportKeyPair](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: Este ejemplo importa una clave pública a EC2. La primera línea almacena el contenido del archivo de clave pública (\*.pub) en la variable. **\$publickey** A continuación, el ejemplo convierte el formato UTF8 del archivo de clave pública en una cadena codificada en Base64 y almacena la cadena convertida en la variable. **\$pkbase64** En la última línea, la clave pública convertida se importa a EC2. Como resultado, el cmdlet devuelve la huella digital y el nombre de la clave.

```
$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
```

Salida:

```
KeyFingerprint                                KeyName
-----
do:d0:15:8f:79:97:12:be:00:fd:df:31:z3:b1:42:z1 Example-user-key
```

- Para obtener información sobre la API, consulte la referencia del [ImportKeyPair AWS Tools for PowerShell](#) cmdlet.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **ImportSnapshot** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar ImportSnapshot.

### CLI

#### AWS CLI

Para importar una instantánea

En el siguiente `import-snapshot` ejemplo, se importa el disco especificado como una instantánea.

```
aws ec2 import-snapshot \
  --description "My server VMDK" \
  --disk-container Format=VMDK,UserBucket={S3Bucket=my-import-bucket,S3Key=vms/
my-server-vm.vmdk}
```

Salida:

```
{
  "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 obtener más información sobre la API, consulte [ImportSnapshot](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: Este ejemplo importa una imagen de disco de máquina virtual con el formato «VMDK» a una instantánea de Amazon EBS. El ejemplo requiere un rol de servicio de importación de máquinas virtuales con el nombre predeterminado «vmimport», con una política que permita a Amazon EC2 acceder al bucket especificado, como se explica en el **VM Import Prerequisites** tema de <http://docs.aws.amazon.com/2/latest/VM.html>. AWSEC WindowsGuide ImportPrerequisites Para usar un rol personalizado, especifique el nombre del rol mediante el parámetro. **-RoleName**

```
$parms = @{
```



```

"ClientToken"="idempotencyToken"
>Description="Disk Image Import"
>DiskContainer_Description" = "Data disk"
>DiskContainer_Format" = "VMDK"
>DiskContainer_S3Bucket" = "myVirtualMachineImages"
>DiskContainer_S3Key" = "datadiskimage.vmdk"
}

Import-EC2Snapshot @parms

```

Salida:

Description	ImportTaskId	SnapshotTaskDetail
-----	-----	-----
Disk Image Import	import-snap-abcdefgh	Amazon.EC2.Model.SnapshotTaskDetail

- Para obtener más información sobre la API, consulte [ImportSnapshot](#) la referencia de AWS Tools for PowerShell cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **ModifyCapacityReservation** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `ModifyCapacityReservation`.

CLI

AWS CLI

Ejemplo 1: Para cambiar el número de instancias reservadas por una reserva de capacidad existente

El siguiente `modify-capacity-reservation` ejemplo cambia el número de instancias para las que la reserva de capacidad reserva capacidad.

```
aws ec2 modify-capacity-reservation \
```

```
--capacity-reservation-id cr-1234abcd56EXAMPLE \  
--instance-count 5
```

Salida:

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

Ejemplo 2: Para cambiar la fecha y la hora de finalización de una reserva de capacidad existente

El siguiente `modify-capacity-reservation` ejemplo modifica una reserva de capacidad existente para que finalice en la fecha y hora especificadas.

```
aws ec2 modify-capacity-reservation \  
  --capacity-reservation-id cr-1234abcd56EXAMPLE \  
  --end-date-type limited \  
  --end-date 2019-08-31T23:59:59Z
```

Para obtener más información, consulte [Modificación de una reserva de capacidad](#) en la Guía del usuario de Amazon Elastic Compute Cloud para instancias de Linux.

- Para obtener más información sobre la API, consulte [ModifyCapacityReservation](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: Este ejemplo modifica el `CapacityReservationId` `cr-0c1f2345db6f7cdba` cambiando el recuento de instancias a 1

```
Edit-EC2CapacityReservation -CapacityReservationId cr-0c1f2345db6f7cdba -  
InstanceCount 1
```

Salida:

```
True
```

- Para obtener información [ModifyCapacityReservation](#) sobre AWS Tools for PowerShell la API, consulte la referencia de cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **ModifyHosts** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `ModifyHosts`.

### CLI

#### AWS CLI

Ejemplo 1: Para habilitar la ubicación automática de un host dedicado

El siguiente `modify-hosts` ejemplo habilita la ubicación automática de un host dedicado para que acepte cualquier lanzamiento de instancia no segmentado que coincida con su configuración de tipo de instancia.

```
aws ec2 modify-hosts \  
  --host-id h-06c2f189b4EXAMPLE \  
  --auto-placement on
```

Salida:

```
{  
  "Successful": [  
    "h-06c2f189b4EXAMPLE"  
  ],  
  "Unsuccessful": []  
}
```

Ejemplo 2: Para habilitar la recuperación de un host dedicado

El siguiente `modify-hosts` ejemplo permite la recuperación del host dedicado especificado.

```
aws ec2 modify-hosts \  
  --host-id h-06c2f189b4EXAMPLE \  
  --host-recovery on
```

**Salida:**

```
{
  "Successful": [
    "h-06c2f189b4EXAMPLE"
  ],
  "Unsuccessful": []
}
```

Para obtener más información, consulte [Modificación de la ubicación automática de un host dedicado](#) en la Guía del usuario de Amazon Elastic Compute Cloud para instancias de Linux.

- Para obtener más información sobre la API, consulte [ModifyHosts](#) la Referencia de AWS CLI comandos.

**PowerShell****Herramientas para PowerShell**

Ejemplo 1: Este ejemplo modifica la AutoPlacement configuración a desactivada para el host dedicado h-01e23f4cd567890f3

```
Edit-EC2Host -HostId h-03e09f8cd681609f3 -AutoPlacement off
```

**Salida:**

```
Successful          Unsuccessful
-----
{h-01e23f4cd567890f3} {}
```

- Para obtener más [ModifyHosts](#) información AWS Tools for PowerShell sobre la API, consulte la referencia de cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

**Úselo `ModifyIdFormat` con un AWS SDK o CLI**

En los siguientes ejemplos de código, se muestra cómo utilizar `ModifyIdFormat`.

## CLI

### AWS CLI

Para habilitar el formato de ID más largo para un recurso

El siguiente `modify-id-format` ejemplo habilita el formato de ID más largo para el tipo de `instance` recurso.

```
aws ec2 modify-id-format \  
  --resource instance \  
  --use-long-ids
```

Para deshabilitar el formato de ID más largo de un recurso

El siguiente `modify-id-format` ejemplo deshabilita el formato de ID más largo para el tipo `instance` de recurso.

```
aws ec2 modify-id-format \  
  --resource instance \  
  --no-use-long-ids
```

El siguiente `modify-id-format` ejemplo habilita el formato de ID más largo para todos los tipos de recursos compatibles que estén dentro de su período de suscripción.

```
aws ec2 modify-id-format \  
  --resource all-current \  
  --use-long-ids
```

- Para obtener más información sobre la API, consulte [ModifyIdFormat](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: Este ejemplo habilita el formato de ID más largo para el tipo de recurso especificado.

```
Edit-EC2IdFormat -Resource instance -UseLongId $true
```

Ejemplo 2: Este ejemplo deshabilita el formato de ID más largo para el tipo de recurso especificado.

```
Edit-EC2IdFormat -Resource instance -UseLongId $false
```

- Para obtener más información sobre la API, consulte la referencia [ModifyIdFormat](#) de AWS Tools for PowerShell cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **ModifyImageAttribute** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `ModifyImageAttribute`.

### CLI

#### AWS CLI

Ejemplo 1: Para hacer pública una AMI

En el siguiente `modify-instance-attribute` ejemplo, se hace pública la AMI especificada.

```
aws ec2 modify-image-attribute \  
  --image-id ami-5731123e \  
  --launch-permission "Add=[{Group=all}]"
```

Este comando no genera ninguna salida.

Ejemplo 2: Para hacer que una AMI sea privada

En el siguiente `modify-instance-attribute` ejemplo, se convierte en privada la AMI especificada.

```
aws ec2 modify-image-attribute \  
  --image-id ami-5731123e \  
  --launch-permission "Remove=[{Group=all}]"
```

Este comando no genera ninguna salida.

Ejemplo 3: conceder el permiso de lanzamiento a una AWS cuenta

En el siguiente `modify-instance-attribute` ejemplo, se conceden permisos de lanzamiento a la AWS cuenta especificada.

```
aws ec2 modify-image-attribute \  
  --image-id ami-5731123e \  
  --launch-permission "Add=[{UserId=123456789012}]"
```

Este comando no genera ninguna salida.

Ejemplo 4: Para eliminar el permiso de lanzamiento de una AWS cuenta

En el siguiente `modify-instance-attribute` ejemplo, se eliminan los permisos de lanzamiento de la AWS cuenta especificada.

```
aws ec2 modify-image-attribute \  
  --image-id ami-5731123e \  
  --launch-permission "Remove=[{UserId=123456789012}]"
```

- Para obtener más información sobre la API, consulte [ModifyImageAttribute](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: en este ejemplo se actualiza la descripción de la AMI especificada.

```
Edit-EC2ImageAttribute -ImageId ami-12345678 -Description "New description"
```

Ejemplo 2: en este ejemplo se hace pública la AMI (por ejemplo, para que cualquiera Cuenta de AWS pueda utilizarla).

```
Edit-EC2ImageAttribute -ImageId ami-12345678 -Attribute launchPermission -  
OperationType add -UserGroup all
```

Ejemplo 3: Este ejemplo hace que la AMI sea privada (por ejemplo, para que solo usted, como propietario, pueda usarla).

```
Edit-EC2ImageAttribute -ImageId ami-12345678 -Attribute launchPermission -
OperationType remove -UserGroup all
```

Ejemplo 4: en este ejemplo se concede el permiso de lanzamiento a la persona especificada Cuenta de AWS.

```
Edit-EC2ImageAttribute -ImageId ami-12345678 -Attribute launchPermission -
OperationType add -UserId 111122223333
```

Ejemplo 5: en este ejemplo se elimina el permiso de lanzamiento del especificado Cuenta de AWS.

```
Edit-EC2ImageAttribute -ImageId ami-12345678 -Attribute launchPermission -
OperationType remove -UserId 111122223333
```

- Para obtener más información sobre la API, consulte [ModifyImageAttribute](#) la referencia del AWS Tools for PowerShell cmdlet.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **ModifyInstanceAttribute** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `ModifyInstanceAttribute`.

### CLI

#### AWS CLI

Ejemplo 1: Para modificar el tipo de instancia

El siguiente `modify-instance-attribute` ejemplo modifica el tipo de instancia de la instancia especificada. La instancia debe tener el estado `stopped`.

```
aws ec2 modify-instance-attribute \  
  --instance-id i-1234567890abcdef0 \  
  --attribute-name instanceType --value t2.micro
```



```
--instance-type "{\"Value\": \"m1.small\"}"
```

Este comando no genera ninguna salida.

Ejemplo 2: Para habilitar una red mejorada en una instancia

El siguiente `modify-instance-attribute` ejemplo habilita una red mejorada para la instancia especificada. La instancia debe tener el estado `stopped`.

```
aws ec2 modify-instance-attribute \  
  --instance-id i-1234567890abcdef0 \  
  --sriov-net-support simple
```

Este comando no genera ninguna salida.

Ejemplo 3: Para modificar el `sourceDestCheck` atributo

El siguiente `modify-instance-attribute` ejemplo establece el `sourceDestCheck` atributo de la instancia especificada en `true`. La instancia debe estar en una VPC.

```
aws ec2 modify-instance-attribute --instance-id i-1234567890abcdef0 --source-  
dest-check "{\"Value\": true}"
```

Este comando no genera ninguna salida.

Ejemplo 4: Para modificar el `deleteOnTermination` atributo del volumen raíz

El siguiente `modify-instance-attribute` ejemplo establece el `deleteOnTermination` atributo del volumen raíz de la instancia respaldada por Amazon EBS especificada en `false`. De forma predeterminada, este atributo es `true` para el volumen raíz.

Comando:

```
aws ec2 modify-instance-attribute \  
  --instance-id i-1234567890abcdef0 \  
  --block-device-mappings "[{\"DeviceName\": \"/dev/sda1\", \"Ebs\": \  
{\"DeleteOnTermination\": false}]"
```

Este comando no genera ninguna salida.

Ejemplo 5: Para modificar los datos de usuario adjuntos a una instancia

En el siguiente `modify-instance-attribute` ejemplo, se agrega el contenido del archivo `UserData.txt` como el `UserData` de la instancia especificada.

Contenido del archivo original `UserData.txt`:

```
#!/bin/bash
yum update -y
service httpd start
chkconfig httpd on
```

El contenido del archivo debe estar codificado en base64. El primer comando convierte el archivo de texto a base64 y lo guarda como un archivo nuevo.

Versión del comando para Linux/macOS:

```
base64 UserData.txt > UserData.base64.txt
```

Este comando no genera ninguna salida.

Versión del comando para Windows:

```
certutil -encode UserData.txt tmp.b64 && findstr /v /c:- tmp.b64 >
UserData.base64.txt
```

Salida:

```
Input Length = 67
Output Length = 152
CertUtil: -encode command completed successfully.
```

Ahora puede hacer referencia a ese archivo en el comando CLI siguiente:

```
aws ec2 modify-instance-attribute \
  --instance-id=i-09b5a14dbca622e76 \
  --attribute userData --value file://UserData.base64.txt
```

Este comando no genera ninguna salida.

Para obtener más información, consulte [Datos de usuario y AWS CLI](#) en la Guía del usuario de EC2.

- Para obtener más información sobre la API, consulte [ModifyInstanceAttribute](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: en este ejemplo se modifica el tipo de instancia de la instancia especificada.

```
Edit-EC2InstanceAttribute -InstanceId i-12345678 -InstanceType m3.medium
```

Ejemplo 2: Este ejemplo permite mejorar las redes para la instancia especificada, especificando «simple» como el valor del parámetro de soporte de red de virtualización de E/S de raíz única (SR-IOV), -.. SriovNetSupport

```
Edit-EC2InstanceAttribute -InstanceId i-12345678 -SriovNetSupport "simple"
```

Ejemplo 3: En este ejemplo se modifican los grupos de seguridad de la instancia especificada. La instancia debe estar en una VPC. Debe especificar el ID de cada grupo de seguridad, no su nombre.

```
Edit-EC2InstanceAttribute -InstanceId i-12345678 -Group @( "sg-12345678",  
"sg-45678901" )
```

Ejemplo 4: Este ejemplo permite la optimización de E/S de EBS para la instancia especificada. Esta función no está disponible en todos los tipos de instancias. Se aplican cargos de uso adicionales al utilizar una instancia optimizada para EBS.

```
Edit-EC2InstanceAttribute -InstanceId i-12345678 -EbsOptimized $true
```

Ejemplo 5: Este ejemplo permite comprobar el origen y el destino de la instancia especificada. Para que una instancia de NAT realice la NAT, el valor debe ser «falso».

```
Edit-EC2InstanceAttribute -InstanceId i-12345678 -SourceDestCheck $true
```

Ejemplo 6: en este ejemplo se inhabilita la terminación de la instancia especificada.

```
Edit-EC2InstanceAttribute -InstanceId i-12345678 -DisableApiTermination $true
```

Ejemplo 7: En este ejemplo se cambia la instancia especificada para que finalice cuando se inicie el cierre desde la instancia.

```
Edit-EC2InstanceAttribute -InstanceId i-12345678 -
InstanceInitiatedShutdownBehavior terminate
```

- Para obtener más información sobre la API, consulte la referencia [ModifyInstanceAttribute](#) del AWS Tools for PowerShell cmdlet.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **ModifyInstanceCreditSpecification** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `ModifyInstanceCreditSpecification`.

### CLI

#### AWS CLI

Para modificar la opción de crédito por el uso de la CPU de una instancia

En este ejemplo, se modifica la opción de crédito para el uso de la CPU de la instancia especificada en la región especificada a «ilimitada». Las opciones de crédito válidas son «estándar» e «ilimitado».

Comando:

```
aws ec2 modify-instance-credit-specification --instance-credit-specification
"InstanceId=i-1234567890abcdef0,CpuCredits=unlimited"
```

Salida:

```
{
  "SuccessfulInstanceCreditSpecifications": [
    {
      "InstanceId": "i-1234567890abcdef0"
    }
  ]
}
```

```
    ],  
    "UnsuccessfulInstanceCreditSpecifications": []  
  }  
}
```

- Para obtener más información sobre la API, consulte [ModifyInstanceCreditSpecification](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: Esto permite créditos T2 ilimitados, por ejemplo, el i-01234567890abcdef.

```
$Credit = New-Object -TypeName  
    Amazon.EC2.Model.InstanceCreditSpecificationRequest  
$Credit.InstanceId = "i-01234567890abcdef"  
$Credit.CpuCredits = "unlimited"  
Edit-EC2InstanceCreditSpecification -InstanceCreditSpecification $Credit
```

- Para obtener información sobre la API, consulte la referencia de cmdlets. [ModifyInstanceCreditSpecification](#) AWS Tools for PowerShell

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **ModifyNetworkInterfaceAttribute** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `ModifyNetworkInterfaceAttribute`.

### CLI

#### AWS CLI

Para modificar el atributo adjunto de una interfaz de red

Este comando de ejemplo modifica el `attachment` atributo de la interfaz de red especificada.

Comando:

```
aws ec2 modify-network-interface-attribute --network-interface-id eni-686ea200 --attachment AttachmentId=eni-attach-43348162,DeleteOnTermination=false
```

Para modificar el atributo de descripción de una interfaz de red

Este comando de ejemplo modifica el `description` atributo de la interfaz de red especificada.

Comando:

```
aws ec2 modify-network-interface-attribute --network-interface-id eni-686ea200 --description "My description"
```

Para modificar el atributo `GroupSet` de una interfaz de red

Este comando de ejemplo modifica el `groupSet` atributo de la interfaz de red especificada.

Comando:

```
aws ec2 modify-network-interface-attribute --network-interface-id eni-686ea200 --groups sg-903004f8 sg-1a2b3c4d
```

Para modificar el `sourceDestCheck` atributo de una interfaz de red

Este comando de ejemplo modifica el `sourceDestCheck` atributo de la interfaz de red especificada.

Comando:

```
aws ec2 modify-network-interface-attribute --network-interface-id eni-686ea200 --no-source-dest-check
```

- Para obtener más información sobre la API, consulte [ModifyNetworkInterfaceAttribute](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: En este ejemplo se modifica la interfaz de red especificada para que el adjunto especificado se elimine al finalizar.

```
Edit-EC2NetworkInterfaceAttribute -NetworkInterfaceId eni-1a2b3c4d -
Attachment_AttachmentId eni-attach-1a2b3c4d -Attachment_DeleteOnTermination $true
```

Ejemplo 2: este ejemplo modifica la descripción de la interfaz de red especificada.

```
Edit-EC2NetworkInterfaceAttribute -NetworkInterfaceId eni-1a2b3c4d -Description
"my description"
```

Ejemplo 3: Este ejemplo modifica el grupo de seguridad de la interfaz de red especificada.

```
Edit-EC2NetworkInterfaceAttribute -NetworkInterfaceId eni-1a2b3c4d -Groups
sg-1a2b3c4d
```

Ejemplo 4: En este ejemplo, se deshabilita la comprobación del origen y el destino de la interfaz de red especificada.

```
Edit-EC2NetworkInterfaceAttribute -NetworkInterfaceId eni-1a2b3c4d -
SourceDestCheck $false
```

- Para obtener más información sobre la API, consulte la referencia de [ModifyNetworkInterfaceAttribute](#) cmdlets AWS Tools for PowerShell .

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **ModifyReservedInstances** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `ModifyReservedInstances`.

### CLI

#### AWS CLI

Para modificar las instancias reservadas

Este comando de ejemplo mueve una instancia reservada a otra zona de disponibilidad en la misma región.

**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
```

**Salida:**

```
{
  "ReservedInstancesModificationId": "rimod-d3ed4335-b1d3-4de6-ab31-0f13aaf46687"
}
```

Para modificar la plataforma de red de las instancias reservadas

Este comando de ejemplo convierte las instancias reservadas EC2-Classic en 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
```

**Salida:**

```
{
  "ReservedInstancesModificationId": "rimod-82fa9020-668f-4fb6-945d-61537009d291"
}
```

Para obtener más información, consulte [Modificación de instancias reservadas](#) en la Guía del usuario de Amazon EC2.

Para modificar el tamaño de las instancias reservadas

Este comando de ejemplo modifica una instancia reservada que tiene 10 instancias m1.small Linux/UNIX en us-west-1c, de modo que 8 instancias m1.small se convierten en 2 instancias m1.large y las 2 m1.small restantes se convierten en 1 m1.medium en la misma zona de disponibilidad. 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
```

### Salida:

```
{
  "ReservedInstancesModificationId": "rimod-acc5f240-080d-4717-b3e3-1c6b11fa00b6"
}
```

Para obtener más información, consulte [Modificación del tamaño de instancia de sus reservas](#) en la Guía del usuario de Amazon EC2.

- Para obtener más información sobre la API, consulte [ModifyReservedInstances](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: este ejemplo modifica la zona de disponibilidad, el recuento de instancias y la plataforma de las instancias 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 obtener más información sobre la API, consulte la referencia [ModifyReservedInstances](#) del AWS Tools for PowerShell cmdlet.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo `ModifySnapshotAttribute` con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `ModifySnapshotAttribute`.

### CLI

#### AWS CLI

Ejemplo 1: Para modificar un atributo de instantánea

En el siguiente `modify-snapshot-attribute` ejemplo, se actualiza el `createVolumePermission` atributo de la instantánea especificada y se eliminan los permisos de volumen del usuario especificado.

```
aws ec2 modify-snapshot-attribute \  
  --snapshot-id snap-1234567890abcdef0 \  
  --attribute createVolumePermission \  
  --operation-type remove \  
  --user-ids 123456789012
```

Ejemplo 2: Hacer pública una instantánea

En el siguiente `modify-snapshot-attribute` ejemplo, se hace pública la instantánea especificada.

```
aws ec2 modify-snapshot-attribute \  
  --snapshot-id snap-1234567890abcdef0 \  
  --attribute createVolumePermission \  
  --operation-type add \  
  --group-names all
```

- Para obtener más información sobre la API, consulte [ModifySnapshotAttribute](#) la Referencia de AWS CLI comandos.

### PowerShell

#### Herramientas para PowerShell

Ejemplo 1: En este ejemplo, se hace pública la instantánea especificada mediante el establecimiento de su `CreateVolumePermission` atributo.

```
Edit-EC2SnapshotAttribute -SnapshotId snap-12345678 -Attribute
CreateVolumePermission -OperationType Add -GroupName all
```

- Para obtener más información sobre la API, consulte [ModifySnapshotAttribute](#) la referencia del AWS Tools for PowerShell cmdlet.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **ModifySpotFleetRequest** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `ModifySpotFleetRequest`.

### CLI

#### AWS CLI

Para modificar una solicitud de flota de Spot

Este comando de ejemplo actualiza la capacidad objetivo de la solicitud de flota de Spot especificada.

Comando:

```
aws ec2 modify-spot-fleet-request --target-capacity 20 --spot-fleet-request-id
sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE
```

Salida:

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

Este comando de ejemplo reduce la capacidad objetivo de la solicitud de flota de spot especificada sin cerrar, por lo tanto, ninguna instancia de 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
```

Salida:

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

- Para obtener más información sobre la API, consulte [ModifySpotFleetRequest](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: En este ejemplo se actualiza la capacidad objetivo de la solicitud de flota de Spot especificada.

```
Edit-EC2SpotFleetRequest -SpotFleetRequestId sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE -TargetCapacity 10
```

Salida:

```
True
```

- Para obtener más información sobre la API, consulte [ModifySpotFleetRequest](#) la referencia de AWS Tools for PowerShell cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **ModifySubnetAttribute** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `ModifySubnetAttribute`.

## CLI

### AWS CLI

Para cambiar el comportamiento del direccionamiento IPv4 público de una subred

En este ejemplo, se modifica la subnet-1a2b3c4d para especificar que a todas las instancias lanzadas en esta subred se les asigne una dirección IPv4 pública. Si el comando se ejecuta correctamente, no se muestra ningún resultado.

Comando:

```
aws ec2 modify-subnet-attribute --subnet-id subnet-1a2b3c4d --map-public-ip-on-launch
```

Para cambiar el comportamiento del direccionamiento IPv6 de una subred

En este ejemplo, se modifica la subnet-1a2b3c4d para especificar que a todas las instancias lanzadas en esta subred se les asigna una dirección IPv6 del rango de la subred.

Comando:

```
aws ec2 modify-subnet-attribute --subnet-id subnet-1a2b3c4d --assign-ipv6-address-on-creation
```

Para obtener más información, consulte Direccionamiento IP en su VPC en la Guía del usuario de AWS Virtual Private Cloud.

- Para obtener más información sobre la API, consulte [ModifySubnetAttribute](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: Este ejemplo habilita el direccionamiento IP público para la subred especificada.

```
Edit-EC2SubnetAttribute -SubnetId subnet-1a2b3c4d -MapPublicIpOnLaunch $true
```

Ejemplo 2: Este ejemplo deshabilita el direccionamiento IP público para la subred especificada.

```
Edit-EC2SubnetAttribute -SubnetId subnet-1a2b3c4d -MapPublicIpOnLaunch $false
```

- Para obtener más información sobre la API, consulte la referencia de [ModifySubnetAttribute](#) [AWS Tools for PowerShell](#) cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **ModifyVolumeAttribute** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `ModifyVolumeAttribute`.

### CLI

#### AWS CLI

Para modificar un atributo de volumen

En este ejemplo se establece el `autoEnableIo` atributo del volumen con el `vol-1234567890abcdef0` identificador en `true`. Si el comando se ejecuta correctamente, no se muestra ningún resultado.

Comando:

```
aws ec2 modify-volume-attribute --volume-id vol-1234567890abcdef0 --auto-enable-io
```

- Para obtener más información sobre la API, consulte [ModifyVolumeAttribute](#) la Referencia de AWS CLI comandos.

### PowerShell

#### Herramientas para PowerShell

Ejemplo 1: En este ejemplo se modifica el atributo especificado del volumen especificado. Las operaciones de E/S del volumen se reanudan automáticamente tras la suspensión debido a la posible incoherencia de los datos.

```
Edit-EC2VolumeAttribute -VolumeId vol-12345678 -AutoEnableIO $true
```

- Para obtener más información sobre la API, consulte la Referencia [ModifyVolumeAttribute](#) de AWS Tools for PowerShell cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **ModifyVpcAttribute** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `ModifyVpcAttribute`.

### CLI

#### AWS CLI

Para modificar el `enableDnsSupport` atributo

En este ejemplo se modifica el `enableDnsSupport` atributo. Este atributo indica si la resolución de DNS está habilitada para la VPC. Si este atributo es `true`, el servidor DNS de Amazon resuelve nombres de host de DNS para sus instancias en sus direcciones IP correspondientes; de lo contrario, no. Si el comando se ejecuta correctamente, no se muestra ningún resultado.

Comando:

```
aws ec2 modify-vpc-attribute --vpc-id vpc-a01106c2 --enable-dns-support "{\"Value\":false}"
```

Para modificar el atributo `enableDnsHostnames`

En este ejemplo se modifica el `enableDnsHostnames` atributo. Este atributo indica si las instancias lanzadas en la VPC obtienen nombres de host DNS. Si el atributo es `true`, las instancias de la VPC obtienen nombres de host de DNS; de lo contrario, no. Si el comando se ejecuta correctamente, no se muestra ningún resultado.

Comando:

```
aws ec2 modify-vpc-attribute --vpc-id vpc-a01106c2 --enable-dns-hostnames
"{\"Value\":false}"
```

- Para obtener información sobre la API, consulte la Referencia [ModifyVpcAttribute](#) de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: Este ejemplo habilita la compatibilidad con los nombres de host DNS para la VPC especificada.

```
Edit-EC2VpcAttribute -VpcId vpc-12345678 -EnableDnsHostnames $true
```

Ejemplo 2: Este ejemplo deshabilita la compatibilidad con los nombres de host DNS para la VPC especificada.

```
Edit-EC2VpcAttribute -VpcId vpc-12345678 -EnableDnsHostnames $false
```

Ejemplo 3: Este ejemplo habilita la compatibilidad con la resolución de DNS para la VPC especificada.

```
Edit-EC2VpcAttribute -VpcId vpc-12345678 -EnableDnsSupport $true
```

Ejemplo 4: Este ejemplo deshabilita la compatibilidad con la resolución de DNS para la VPC especificada.

```
Edit-EC2VpcAttribute -VpcId vpc-12345678 -EnableDnsSupport $false
```

- Para obtener más información sobre la API, consulte la referencia de [ModifyVpcAttribute](#) [AWS Tools for PowerShell](#) cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.



## Úselo `MonitorInstances` con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `MonitorInstances`.

### C++

#### SDK para C++

#### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

```
Aws::EC2::EC2Client ec2Client(clientConfiguration);
Aws::EC2::Model::MonitorInstancesRequest request;
request.AddInstanceIds(instanceId);
request.SetDryRun(true);

auto dry_run_outcome = ec2Client.MonitorInstances(request);
if (dry_run_outcome.IsSuccess()) {
    std::cerr
        << "Failed dry run to enable monitoring on instance. A dry run
should trigger an error."
        <<
        std::endl;
    return false;
}
else if (dry_run_outcome.GetError().GetErrorType()
    != Aws::EC2::EC2Errors::DRY_RUN_OPERATION) {
    std::cerr << "Failed dry run to enable monitoring on instance " <<
        instanceId << ": " << dry_run_outcome.GetError().GetMessage()
<<
        std::endl;
    return false;
}

request.SetDryRun(false);
auto monitorInstancesOutcome = ec2Client.MonitorInstances(request);
if (!monitorInstancesOutcome.IsSuccess()) {
    std::cerr << "Failed to enable monitoring on instance " <<
        instanceId << ": " <<
```

```
        monitorInstancesOutcome.GetError().GetMessage() << std::endl;
    }
    else {
        std::cout << "Successfully enabled monitoring on instance " <<
            instanceId << std::endl;
    }
}
```

- Para obtener más información sobre la API, consulta [MonitorInstances](#) la Referencia AWS SDK for C++ de la API.

## CLI

### AWS CLI

Habilitar el monitoreo detallado para una instancia

Este comando de ejemplo habilita el monitoreo detallado de la instancia especificada.

Comando:

```
aws ec2 monitor-instances --instance-ids i-1234567890abcdef0
```

Salida:

```
{
  "InstanceMonitorings": [
    {
      "InstanceId": "i-1234567890abcdef0",
      "Monitoring": {
        "State": "pending"
      }
    }
  ]
}
```

- Para obtener más información sobre la API, consulta [MonitorInstances](#) la Referencia de AWS CLI comandos.

## JavaScript

### SDK para JavaScript (v3)

#### Note

Hay más información. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

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

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

// Turn on detailed monitoring for the selected instance.
// By default, metrics are sent to Amazon CloudWatch every 5 minutes.
// For a cost you can enable detailed monitoring which sends metrics every
minute.
export const main = async () => {
  const command = new MonitorInstancesCommand({
    InstanceIds: ["INSTANCE_ID"],
  });

  try {
    const { InstanceMonitorings } = await client.send(command);
    const instancesBeingMonitored = InstanceMonitorings.map(
      (im) =>
        ` • Detailed monitoring state for ${im.InstanceId} is
${im.Monitoring.State}.`,
    );
    console.log("Monitoring status:");
    console.log(instancesBeingMonitored.join("\n"));
  } catch (err) {
    console.error(err);
  }
};
```

- Para obtener más información sobre la API, consulta [MonitorInstances](#) la Referencia AWS SDK for JavaScript de la API.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: Este ejemplo permite una supervisión detallada de la instancia especificada.

```
Start-EC2InstanceMonitoring -InstanceId i-12345678
```

Salida:

```
InstanceId      Monitoring
-----
i-12345678     Amazon.EC2.Model.Monitoring
```

- Para obtener más información sobre la API, consulte [MonitorInstances](#) la referencia del AWS Tools for PowerShell cmdlet.

## SAP ABAP

### SDK de SAP ABAP

#### Note

Hay más información al respecto. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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.
  ENTRY.

```

- Para obtener más información sobre la API, consulte [MonitorInstances](#) la referencia sobre la API ABAP del AWS SDK para SAP.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte [Cree recursos de Amazon EC2 mediante un SDK AWS](#). En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **MoveAddressToVpc** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar MoveAddressToVpc.

### CLI

#### AWS CLI

Para mover una dirección a EC2-VPC

En este ejemplo, se mueve la dirección IP elástica 54.123.4.56 a la plataforma EC2-VPC.

## Comando:

```
aws ec2 move-address-to-vpc --public-ip 54.123.4.56
```

## Salida:

```
{
  "Status": "MoveInProgress"
}
```

- Para obtener más información sobre la API, consulte la Referencia de comandos. [MoveAddressToVpcAWS CLI](#)

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: Este ejemplo mueve una instancia EC2 con una dirección IP pública de 12.345.67.89 a la plataforma EC2-VPC en la región EE.UU. Este (Norte de Virginia).

```
Move-EC2AddressToVpc -PublicIp 12.345.67.89 -Region us-east-1
```

Ejemplo 2: este ejemplo canaliza los resultados de un comando al cmdlet. Get-EC2Instance Move-EC2AddressToVpc El Get-EC2Instance comando obtiene una instancia que se especifica mediante el ID de la instancia y, a continuación, devuelve la propiedad de dirección IP pública de la instancia.

```
(Get-EC2Instance -Instance i-12345678).Instances.PublicIpAddress | Move-EC2AddressToVpc
```

- Para obtener más información sobre la API, consulta [MoveAddressToVpc](#) la referencia de AWS Tools for PowerShell cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo `PurchaseHostReservation` con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `PurchaseHostReservation`.

### CLI

#### AWS CLI

Para comprar una reserva de anfitrión dedicada

En este ejemplo, se compra la oferta de reserva de anfitrión dedicado especificada para el host dedicado especificado de su cuenta.

Comando:

```
aws ec2 purchase-host-reservation --offering-id hro-03f707bf363b6b324 --host-id-set h-013abcd2a00cbd123
```

Salida:

```
{
  "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 obtener más información sobre la API, consulte [PurchaseHostReservation](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: Este ejemplo compra la oferta de reservas hro-0c1f23456789d0ab con configuraciones que coinciden con las de su host dedicado h-01e23f4cd567890f1

```
New-EC2HostReservation -OfferingId hro-0c1f23456789d0ab HostIdSet
h-01e23f4cd567890f1
```

#### Salida:

```
ClientToken      :
CurrencyCode     :
Purchase         : {hr-0123f4b5d67bedc89}
TotalHourlyPrice : 1.307
TotalUpfrontPrice : 0.000
```

- Para obtener [PurchaseHostReservation](#) más AWS Tools for PowerShell información sobre la API, consulte la referencia de cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **PurchaseScheduledInstances** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `PurchaseScheduledInstances`.

### CLI

#### AWS CLI

Para comprar una instancia programada

En este ejemplo, se compra una instancia programada.

Comando:

```
aws ec2 purchase-scheduled-instances --purchase-requests file://purchase-
request.json
```



## Purchase-request.json:

```
[
  {
    "PurchaseToken": "eyJ2IjoiMSIsInMiOjEsImMiOi...",
    "InstanceCount": 1
  }
]
```

## Salida:

```
{
  "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 obtener más información sobre la API, consulte la Referencia de comandos. [PurchaseScheduledInstances](#) AWS CLI

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: En este ejemplo se compra una instancia programada.

```
$request = New-Object Amazon.EC2.Model.PurchaseRequest
$request.InstanceCount = 1
$request.PurchaseToken = "eyJ2IjoiMSIsInMiOjEsImMiOi..."
New-EC2ScheduledInstancePurchase -PurchaseRequest $request
```

Salida:

```
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 obtener información sobre la API, consulte [PurchaseScheduledInstances](#) la referencia del AWS Tools for PowerShell cmdlet.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **RebootInstances** con un AWS SDK o CLI


En los siguientes ejemplos de código, se muestra cómo utilizar `RebootInstances`.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en contexto en el siguiente ejemplo de código:

- [Creación y administración de un servicio resiliente](#)

.NET

AWS SDK for .NET

 Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

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

    var response = await _amazonEC2.RebootInstancesAsync(request);
    if (response.HttpStatusCode == System.Net.HttpStatusCode.OK)
    {
        Console.WriteLine("Instances successfully rebooted.");
    }
    else
    {
        Console.WriteLine("Could not reboot one or more instances.");
    }
}
```

Sustituya el perfil por una instancia y reinicie un 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)
{
    await _amazonEc2.ReplaceIamInstanceProfileAssociationAsync(
        new ReplaceIamInstanceProfileAssociationRequest()
        {
            AssociationId = associationId,
            IamInstanceProfile = new IamInstanceProfileSpecification()
            {
                Name = credsProfileName
            }
        });
    // Allow time before resetting.
    Thread.Sleep(25000);
    var instanceReady = false;
    var retries = 5;
    while (retries-- > 0 && !instanceReady)
    {
        await _amazonEc2.RebootInstancesAsync(
            new RebootInstancesRequest(new List<string>() { instanceId }));
        Thread.Sleep(10000);

        var instancesPaginator =
            _amazonSsm.Paginators.DescribeInstanceInformation(
                new DescribeInstanceInformationRequest());
        // Get the entire list using the paginator.
        await foreach (var instance in
            instancesPaginator.InstanceInformationList)
        {
            instanceReady = instance.InstanceId == instanceId;
        }
    }
}
```

```
        if (instanceReady)
        {
            break;
        }
    }
}
Console.WriteLine($"Sending restart command to instance {instanceId}");
await _amazonSsm.SendCommandAsync(
    new SendCommandRequest()
    {
        InstanceIds = new List<string>() { instanceId },
        DocumentName = "AWS-RunShellScript",
        Parameters = new Dictionary<string, List<string>>()
        {
            {"commands", new List<string>() { "cd / && sudo python3
server.py 80" }}
        }
    });
Console.WriteLine($"Restarted the web server on instance {instanceId}");
}
```

- Para obtener más información sobre la API, consulta [RebootInstances](#) la Referencia AWS SDK for .NET de la API.

## C++

### SDK para C++

#### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

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

Aws::EC2::Model::RebootInstancesRequest request;
request.AddInstanceIds(instanceId);
request.SetDryRun(true);
```

```
auto dry_run_outcome = ec2Client.RebootInstances(request);
if (dry_run_outcome.IsSuccess()) {
    std::cerr
        << "Failed dry run to reboot on instance. A dry run should
trigger an error."
        <<
        std::endl;
    return false;
}
else if (dry_run_outcome.GetError().GetErrorType()
    != Aws::EC2::EC2Errors::DRY_RUN_OPERATION) {
    std::cout << "Failed dry run to reboot instance " << instanceId << ": "
        << dry_run_outcome.GetError().GetMessage() << std::endl;
    return false;
}

request.SetDryRun(false);
auto outcome = ec2Client.RebootInstances(request);
if (!outcome.IsSuccess()) {
    std::cout << "Failed to reboot instance " << instanceId << ": " <<
        outcome.GetError().GetMessage() << std::endl;
}
else {
    std::cout << "Successfully rebooted instance " << instanceId <<
        std::endl;
}
}
```

- Para obtener más información sobre la API, consulta [RebootInstances](#) la Referencia AWS SDK for C++ de la API.

## CLI

### AWS CLI

#### Reiniciar una instancia de Amazon EC2

En este ejemplo, se reinicia la instancia especificada. Si el comando se ejecuta correctamente, no se muestra ningún resultado.

Comando:

```
aws ec2 reboot-instances --instance-ids i-1234567890abcdef5
```

Para obtener más información, consulte Reiniciar su instancia en la Guía del usuario de Amazon Elastic Compute Cloud.

- Para obtener más información sobre la API, consulta [RebootInstances](#) la Referencia de AWS CLI comandos.

## JavaScript

### SDK para JavaScript (v3)

#### Note

Hay más información. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

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

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

export const main = async () => {
  const command = new RebootInstancesCommand({
    InstanceIds: ["INSTANCE_ID"],
  });

  try {
    await client.send(command);
    console.log("Instance rebooted successfully.");
  } catch (err) {
    console.error(err);
  }
};
```

- Para obtener más información sobre la API, consulta [RebootInstances](#) la Referencia AWS SDK for JavaScript de la API.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: en este ejemplo se reinicia la instancia especificada.

```
Restart-EC2Instance -InstanceId i-12345678
```

- Para obtener más información sobre la API, consulte la referencia [RebootInstances](#) del AWS Tools for PowerShell cmdlet.

## Python

### SDK para Python (Boto3)

#### Note

Hay más información al respecto. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

```
class AutoScaler:
    """
    Encapsulates Amazon EC2 Auto Scaling and EC2 management actions.
    """

    def __init__(
        self,
        resource_prefix,
        inst_type,
        ami_param,
        autoscaling_client,
        ec2_client,
        ssm_client,
        iam_client,
    ):
        """
        :param resource_prefix: The prefix for naming AWS resources that are
        created by this class.
        :param inst_type: The type of EC2 instance to create, such as t3.micro.
```



```

        :param ami_param: The Systems Manager parameter used to look up the AMI
that is
                created.
        :param autoscaling_client: A Boto3 EC2 Auto Scaling client.
        :param ec2_client: A Boto3 EC2 client.
        :param ssm_client: A Boto3 Systems Manager client.
        :param iam_client: A Boto3 IAM client.
        """
self.inst_type = inst_type
self.ami_param = ami_param
self.autoscaling_client = autoscaling_client
self.ec2_client = ec2_client
self.ssm_client = ssm_client
self.iam_client = iam_client
self.launch_template_name = f"{resource_prefix}-template"
self.group_name = f"{resource_prefix}-group"
self.instance_policy_name = f"{resource_prefix}-pol"
self.instance_role_name = f"{resource_prefix}-role"
self.instance_profile_name = f"{resource_prefix}-prof"
self.bad_creds_policy_name = f"{resource_prefix}-bc-pol"
self.bad_creds_role_name = f"{resource_prefix}-bc-role"
self.bad_creds_profile_name = f"{resource_prefix}-bc-prof"
self.key_pair_name = f"{resource_prefix}-key-pair"

def replace_instance_profile(
    self, instance_id, new_instance_profile_name, profile_association_id
):
    """
    Replaces the profile associated with a running instance. After the
profile is
    replaced, the instance is rebooted to ensure that it uses the new
profile. When
    the instance is ready, Systems Manager is used to restart the Python web
server.

    :param instance_id: The ID of the instance to update.
    :param new_instance_profile_name: The name of the new profile to
associate with
                the specified instance.
    :param profile_association_id: The ID of the existing profile association
for the
                instance.
    """

```

```
try:
    self.ec2_client.replace_iam_instance_profile_association(
        IamInstanceProfile={"Name": new_instance_profile_name},
        AssociationId=profile_association_id,
    )
    log.info(
        "Replaced instance profile for association %s with profile %s.",
        profile_association_id,
        new_instance_profile_name,
    )
    time.sleep(5)
    inst_ready = False
    tries = 0
    while not inst_ready:
        if tries % 6 == 0:
            self.ec2_client.reboot_instances(InstanceIds=[instance_id])
            log.info(
                "Rebooting instance %s and waiting for it to be
ready.",
                instance_id,
            )
            tries += 1
            time.sleep(10)
            response = self.ssm_client.describe_instance_information()
            for info in response["InstanceInformationList"]:
                if info["InstanceId"] == instance_id:
                    inst_ready = True
    self.ssm_client.send_command(
        InstanceIds=[instance_id],
        DocumentName="AWS-RunShellScript",
        Parameters={"commands": ["cd / && sudo python3 server.py 80"]},
    )
    log.info("Restarted the Python web server on instance %s.",
instance_id)
    except ClientError as err:
        raise AutoScalerError(
            f"Couldn't replace instance profile for association
{profile_association_id}: {err}"
        )
```

- Para obtener más información sobre la API, consulta [RebootInstances](#) la AWS Referencia de API de SDK for Python (Boto3).

## Rust

### SDK para Rust

#### Note

Hay más información al respecto. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

```
async fn reboot_instance(client: &Client, id: &str) -> Result<(), Error> {
    println!("Rebooting instance.");

    client.reboot_instances().instance_ids(id).send().await?;

    client
        .wait_until_instance_stopped()
        .instance_ids(id)
        .wait(Duration::from_secs(60))
        .await?;
    let wait_status_ok = client
        .wait_until_instance_status_ok()
        .instance_ids(id)
        .wait(Duration::from_secs(60))
        .await;

    match wait_status_ok {
        Ok(_) => println!("Rebooted instance {id}, it is started with status
OK."),
        Err(err) => return Err(err.into()),
    }

    Ok(())
}
```

- Para obtener más información sobre la API, consulta [RebootInstances](#) la referencia sobre la API de AWS SDK para Rust.

## SAP ABAP

## SDK de SAP ABAP

 Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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 obtener más información sobre la API, consulte [RebootInstances](#) la referencia sobre la API ABAP del AWS SDK para SAP.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte [Cree recursos de Amazon EC2 mediante un SDK AWS](#). En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **RegisterImage** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar RegisterImage.

### CLI

#### AWS CLI

Ejemplo 1: Para registrar una AMI mediante un archivo de manifiesto

En el siguiente `register-image` ejemplo, se registra una AMI mediante el archivo de manifiesto especificado en Amazon S3.

```
aws ec2 register-image \  
  --name my-image \  
  --image-location my-s3-bucket/myimage/image.manifest.xml
```

Salida:

```
{  
  "ImageId": "ami-1234567890EXAMPLE"  
}
```

Para obtener más información, consulte [Imágenes de máquina de Amazon \(AMI\)](#) en la Guía del usuario de Amazon EC2.

Ejemplo 2: Para registrar una AMI mediante una instantánea de un dispositivo raíz

El siguiente `register-image` ejemplo registra una AMI utilizando la instantánea especificada de un volumen raíz de EBS como dispositivo `/dev/xvda`. El mapeo del dispositivo de bloques también incluye un volumen EBS vacío 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}
```

Salida:

```
{  
  "ImageId": "ami-1a2b3c4d5eEXAMPLE"  
}
```

Para obtener más información, consulte [Imágenes de máquina de Amazon \(AMI\)](#) en la Guía del usuario de Amazon EC2.

- Para obtener más información sobre la API, consulte la Referencia [RegisterImage](#) de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: en este ejemplo se registra una AMI mediante el archivo de manifiesto especificado en Amazon S3.

```
Register-EC2Image -ImageLocation my-s3-bucket/my-web-server-ami/  
image.manifest.xml -Name my-web-server-ami
```

- Para obtener más información sobre la API, consulte [RegisterImage](#) la referencia de AWS Tools for PowerShell cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo `RejectVpcPeeringConnection` con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `RejectVpcPeeringConnection`.

### CLI

#### AWS CLI

Para rechazar una conexión de emparejamiento de VPC

En este ejemplo, se rechaza la solicitud de conexión de emparejamiento de VPC especificada.

Comando:

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

Salida:

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

- Para obtener más información sobre la API, consulte la Referencia [RejectVpcPeeringConnection](#) de AWS CLI comandos.

### PowerShell

#### Herramientas para PowerShell

Ejemplo 1: El ejemplo anterior deniega la solicitud del identificador de solicitud `VpcPeering pcx-01a2b3ce45fe67eb8`

```
Deny-EC2VpcPeeringConnection -VpcPeeringConnectionId pcx-01a2b3ce45fe67eb8
```

- Para obtener más [RejectVpcPeeringConnection](#) información AWS Tools for PowerShell sobre la API, consulte la referencia del cmdlet.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **ReleaseAddress** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `ReleaseAddress`.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en contexto en el siguiente ejemplo de código:

- [Comience a utilizar instancias](#)

### .NET

#### AWS SDK for .NET

#### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

```
/// <summary>
/// Release an Elastic IP address.
/// </summary>
/// <param name="allocationId">The allocation Id of the Elastic IP address.</
param>
/// <returns>A Boolean value indicating the success of the action.</returns>
public async Task<bool> ReleaseAddress(string allocationId)
{
    var request = new ReleaseAddressRequest
    {
        AllocationId = allocationId
    };

    var response = await _amazonEC2.ReleaseAddressAsync(request);
    return response.HttpStatusCode == HttpStatusCode.OK;
}
```

- Para obtener más información sobre la API, consulta [ReleaseAddress](#) la Referencia AWS SDK for .NET de la API.



## Bash

## AWS CLI con el script Bash

 Note

Hay más información. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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
}

```

Las funciones de utilidad utilizadas en este ejemplo.

```

#####
# 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 obtener más información sobre la API, consulta [ReleaseAddress](#) la Referencia de AWS CLI comandos.

## C++

### SDK para C++

#### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

```
Aws::EC2::EC2Client ec2(clientConfiguration);

Aws::EC2::Model::ReleaseAddressRequest request;
request.SetAllocationId(allocationID);

auto outcome = ec2.ReleaseAddress(request);
if (!outcome.IsSuccess()) {
    std::cerr << "Failed to release Elastic IP address " <<
                allocationID << ":" << outcome.GetError().GetMessage() <<
                std::endl;
}
else {
    std::cout << "Successfully released Elastic IP address " <<
                allocationID << std::endl;
}
}
```

- Para obtener más información sobre la API, consulta [ReleaseAddress](#) la Referencia AWS SDK for C++ de la API.

## CLI

### AWS CLI

#### Liberar direcciones IP elásticas para EC2-Classic

En este ejemplo, se libera una dirección IP elástica para su uso con instancias en EC2-Classic. Si el comando se ejecuta correctamente, no se muestra ningún resultado.

Comando:

```
aws ec2 release-address --public-ip 198.51.100.0
```

## Liberar una dirección IP elástica para EC2-VPC

En este ejemplo, se libera una dirección IP elástica para su uso con instancias en una VPC. Si el comando se ejecuta correctamente, no se muestra ningún resultado.

Comando:

```
aws ec2 release-address --allocation-id eipalloc-64d5890a
```

- Para obtener más información sobre la API, consulta [ReleaseAddress](#) la Referencia de AWS CLI comandos.

## Java

### SDK para Java 2.x

#### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

```
public static void releaseEC2Address(Ec2Client ec2, String allocId) {
    try {
        ReleaseAddressRequest request = ReleaseAddressRequest.builder()
            .allocationId(allocId)
            .build();

        ec2.releaseAddress(request);
        System.out.println("Successfully released Elastic IP address " +
allocId);
    } catch (Ec2Exception e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
}
```

- Para obtener más información sobre la API, consulta [ReleaseAddress](#) la Referencia AWS SDK for Java 2.x de la API.

## JavaScript

### SDK para JavaScript (v3)

#### Note

Hay más información. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

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

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

export const main = async () => {
  const command = new ReleaseAddressCommand({
    // You can also use PublicIp, but that is for EC2 classic which is being
    // retired.
    AllocationId: "ALLOCATION_ID",
  });

  try {
    await client.send(command);
    console.log("Successfully released address.");
  } catch (err) {
    console.error(err);
  }
};
```

- Para obtener más información sobre la API, consulta [ReleaseAddress](#) la Referencia AWS SDK for JavaScript de la API.

## Kotlin

### SDK para Kotlin

#### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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 obtener más información sobre la API, consulta [ReleaseAddress](#) la referencia sobre el AWS SDK para la API de Kotlin.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: en este ejemplo se publica la dirección IP elástica especificada para las instancias de una VPC.

```
Remove-EC2Address -AllocationId eipalloc-12345678 -Force
```

Ejemplo 2: En este ejemplo, se publica la dirección IP elástica especificada para las instancias de EC2-Classic.

```
Remove-EC2Address -PublicIp 198.51.100.2 -Force
```

- Para obtener más información sobre la API, consulte la referencia de [ReleaseAddress AWS Tools for PowerShell](#) cmdlets.

## Python

### SDK para Python (Boto3)

#### Note

Hay más información al respecto. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

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

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

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

    def release(self):
        """
        Releases an Elastic IP address. After the Elastic IP address is released,
        it can no longer be used.
        """
```



```
if self.elastic_ip is None:
    logger.info("No Elastic IP to release.")
    return

try:
    self.elastic_ip.release()
except ClientError as err:
    logger.error(
        "Couldn't release Elastic IP address %s. Here's why: %s: %s",
        self.elastic_ip.allocation_id,
        err.response["Error"]["Code"],
        err.response["Error"]["Message"],
    )
    raise
```

- Para obtener más información sobre la API, consulta [ReleaseAddress](#) la AWS Referencia de API de SDK for Python (Boto3).

## Ruby

### SDK para Ruby

#### Note

Hay más información al respecto. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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)
  return true
rescue StandardError => e
  puts("Error releasing Elastic IP address: #{e.message}")
  return false
end

```

- Para obtener más información sobre la API, consulta [ReleaseAddress](#) la Referencia AWS SDK for Ruby de la API.

## SAP ABAP

### SDK de SAP ABAP

#### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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 obtener más información sobre la API, consulte [ReleaseAddress](#) la referencia sobre la API ABAP del AWS SDK para SAP.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte [Cree recursos de Amazon EC2 mediante un SDK AWS](#). En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **ReleaseHosts** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `ReleaseHosts`.

### CLI

#### AWS CLI

Para liberar un host dedicado de tu cuenta

Para liberar un host dedicado de tu cuenta. Las instancias que se encuentran en el host deben detenerse o cancelarse antes de que se pueda liberar el host.

Comando:

```
aws ec2 release-hosts --host-id=h-0029d6e3cacf1b3da
```

Salida:

```
{
  "Successful": [
    "h-0029d6e3cacf1b3da"
  ],
  "Unsuccessful": []
}
```

- Para obtener más información sobre la API, consulte [ReleaseHosts](#) la Referencia de AWS CLI comandos.

### PowerShell

#### Herramientas para PowerShell

Ejemplo 1: En este ejemplo se publica el ID de host indicado `h-0badafd1dcb2f3456`

```
Remove-EC2Host -HostId h-0badafd1dcb2f3456
```

**Salida:**

```

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 obtener información [ReleaseHosts](#) sobre AWS Tools for PowerShell la API, consulte la referencia del cmdlet.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **ReplaceIamInstanceProfileAssociation** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `ReplaceIamInstanceProfileAssociation`.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en contexto en el siguiente ejemplo de código:

- [Creación y administración de un servicio resiliente](#)

### .NET

#### AWS SDK for .NET

##### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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)
{
    await _amazonEc2.ReplaceIamInstanceProfileAssociationAsync(
        new ReplaceIamInstanceProfileAssociationRequest()
        {
            AssociationId = associationId,
            IamInstanceProfile = new IamInstanceProfileSpecification()
            {
                Name = credsProfileName
            }
        });
    // Allow time before resetting.
    Thread.Sleep(25000);
    var instanceReady = false;
    var retries = 5;
    while (retries-- > 0 && !instanceReady)
    {
        await _amazonEc2.RebootInstancesAsync(
            new RebootInstancesRequest(new List<string>() { instanceId }));
        Thread.Sleep(10000);

        var instancesPaginator =
            _amazonSsm.Paginators.DescribeInstanceInformation(
                new DescribeInstanceInformationRequest());
        // Get the entire list using the paginator.
        await foreach (var instance in
            instancesPaginator.InstanceInformationList)
        {
            instanceReady = instance.InstanceId == instanceId;
        }
    }
}
```

```
        if (instanceReady)
        {
            break;
        }
    }
}
Console.WriteLine($"Sending restart command to instance {instanceId}");
await _amazonSsm.SendCommandAsync(
    new SendCommandRequest()
    {
        InstanceIds = new List<string>() { instanceId },
        DocumentName = "AWS-RunShellScript",
        Parameters = new Dictionary<string, List<string>>()
        {
            {"commands", new List<string>() { "cd / && sudo python3
server.py 80" }}
        }
    });
Console.WriteLine($"Restarted the web server on instance {instanceId}");
}
```

- Para obtener más información sobre la API, consulta [ReplacelamInstanceProfileAssociation](#) la Referencia AWS SDK for .NET de la API.

## CLI

### AWS CLI

Reemplazar un perfil de instancia de IAM para una instancia

En este ejemplo, se reemplaza el perfil de instancia de IAM, representado por la asociación `iip-assoc-060bae234aac2e7fa`, con el perfil de instancia de IAM denominado `AdminRole`.

```
aws ec2 replace-iam-instance-profile-association \  
  --iam-instance-profile Name=AdminRole \  
  --association-id iip-assoc-060bae234aac2e7fa
```

Salida:

```
{
```

```

    "IamInstanceProfileAssociation": {
      "InstanceId": "i-087711ddaf98f9489",
      "State": "associating",
      "AssociationId": "iip-assoc-0b215292fab192820",
      "IamInstanceProfile": {
        "Id": "AIPAJLNLDX3AMYZNWYYAY",
        "Arn": "arn:aws:iam::123456789012:instance-profile/AdminRole"
      }
    }
  }
}

```

- Para obtener más información sobre la API, consulta [ReplacelamInstanceProfileAssociation](#) la Referencia de AWS CLI comandos.

## JavaScript

### SDK para JavaScript (v3)

#### Note

Hay más información. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

```

await retry({ intervalInMs: 1000, maxRetries: 30 }, () =>
  ec2Client.send(
    new ReplaceIamInstanceProfileAssociationCommand({
      AssociationId: state.instanceProfileAssociationId,
      IamInstanceProfile: { Name: NAMES.ssmOnlyInstanceProfileName },
    }),
  ),
);

```

- Para obtener más información sobre la API, consulta [ReplacelamInstanceProfileAssociation](#) la Referencia AWS SDK for JavaScript de la API.

## Python

### SDK para Python (Boto3)

#### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

Este ejemplo reemplaza el perfil de instancia de una instancia en ejecución, reinicia la instancia y envía un comando a la instancia una vez iniciada.

```
class AutoScaler:
    """
    Encapsulates Amazon EC2 Auto Scaling and EC2 management actions.
    """

    def __init__(
        self,
        resource_prefix,
        inst_type,
        ami_param,
        autoscaling_client,
        ec2_client,
        ssm_client,
        iam_client,
    ):
        """
        :param resource_prefix: The prefix for naming AWS resources that are
        created by this class.
        :param inst_type: The type of EC2 instance to create, such as t3.micro.
        :param ami_param: The Systems Manager parameter used to look up the AMI
        that is
                created.
        :param autoscaling_client: A Boto3 EC2 Auto Scaling client.
        :param ec2_client: A Boto3 EC2 client.
        :param ssm_client: A Boto3 Systems Manager client.
        :param iam_client: A Boto3 IAM client.
        """
        self.inst_type = inst_type
        self.ami_param = ami_param
```



```

self.autoscaling_client = autoscaling_client
self.ec2_client = ec2_client
self.ssm_client = ssm_client
self.iam_client = iam_client
self.launch_template_name = f"{resource_prefix}-template"
self.group_name = f"{resource_prefix}-group"
self.instance_policy_name = f"{resource_prefix}-pol"
self.instance_role_name = f"{resource_prefix}-role"
self.instance_profile_name = f"{resource_prefix}-prof"
self.bad_creds_policy_name = f"{resource_prefix}-bc-pol"
self.bad_creds_role_name = f"{resource_prefix}-bc-role"
self.bad_creds_profile_name = f"{resource_prefix}-bc-prof"
self.key_pair_name = f"{resource_prefix}-key-pair"

def replace_instance_profile(
    self, instance_id, new_instance_profile_name, profile_association_id
):
    """
    Replaces the profile associated with a running instance. After the
    profile is
    replaced, the instance is rebooted to ensure that it uses the new
    profile. When
    the instance is ready, Systems Manager is used to restart the Python web
    server.

    :param instance_id: The ID of the instance to update.
    :param new_instance_profile_name: The name of the new profile to
    associate with
                                the specified instance.
    :param profile_association_id: The ID of the existing profile association
    for the
                                instance.
    """
    try:
        self.ec2_client.replace_iam_instance_profile_association(
            IamInstanceProfile={"Name": new_instance_profile_name},
            AssociationId=profile_association_id,
        )
        log.info(
            "Replaced instance profile for association %s with profile %s.",
            profile_association_id,
            new_instance_profile_name,
        )

```

```

        time.sleep(5)
        inst_ready = False
        tries = 0
        while not inst_ready:
            if tries % 6 == 0:
                self.ec2_client.reboot_instances(InstanceIds=[instance_id])
                log.info(
                    "Rebooting instance %s and waiting for it to be
ready.",
                    instance_id,
                )
            tries += 1
            time.sleep(10)
            response = self.ssm_client.describe_instance_information()
            for info in response["InstanceInformationList"]:
                if info["InstanceId"] == instance_id:
                    inst_ready = True
        self.ssm_client.send_command(
            InstanceIds=[instance_id],
            DocumentName="AWS-RunShellScript",
            Parameters={"commands": ["cd / && sudo python3 server.py 80"]},
        )
        log.info("Restarted the Python web server on instance %s.",
instance_id)
    except ClientError as err:
        raise AutoScalerError(
            f"Couldn't replace instance profile for association
{profile_association_id}: {err}")
    )

```

- Para obtener más información sobre la API, consulta [ReplacelamInstanceProfileAssociation](#) la AWS Referencia de API de SDK for Python (Boto3).

Para obtener una lista completa de las guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

# Úselo `ReplaceNetworkAclAssociation` con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `ReplaceNetworkAclAssociation`.

## CLI

### AWS CLI

Para reemplazar la ACL de red asociada a una subred

En este ejemplo, se asocia la ACL de red especificada a la subred de la asociación de ACL de red especificada.

Comando:

```
aws ec2 replace-network-acl-association --association-id aclassoc-e5b95c8c --network-acl-id acl-5fb85d36
```

Salida:

```
{
  "NewAssociationId": "aclassoc-3999875b"
}
```

- Para obtener más información sobre la API, consulte [ReplaceNetworkAclAssociation](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: Este ejemplo asocia la ACL de red especificada con la subred de la asociación de ACL de red especificada.

```
Set-EC2NetworkAclAssociation -NetworkAclId acl-12345678 -AssociationId aclassoc-1a2b3c4d
```

Salida:

```
aclassoc-87654321
```

- Para obtener más información sobre la API, consulte la referencia [ReplaceNetworkAclAssociation](#) de AWS Tools for PowerShell cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **ReplaceNetworkAclEntry** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `ReplaceNetworkAclEntry`.

### CLI

#### AWS CLI

Para reemplazar una entrada de ACL de red

Este ejemplo reemplaza una entrada para la ACL de red especificada. La nueva regla 100 permite la entrada de tráfico desde 203.0.113.12/24 por el puerto UDP 53 (DNS) a cualquier subred asociada.

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 obtener más información sobre la API, consulte la Referencia de comandos. [ReplaceNetworkAclEntry](#) AWS CLI

### PowerShell

#### Herramientas para PowerShell

Ejemplo 1: Este ejemplo reemplaza la entrada especificada para la ACL de red especificada. La nueva regla permite el tráfico entrante desde la dirección especificada a cualquier subred asociada.

```
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 obtener más información sobre la API, consulte la referencia de [ReplaceNetworkAclEntry AWS Tools for PowerShell](#) cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **ReplaceRoute** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `ReplaceRoute`.

### CLI

#### AWS CLI

Para reemplazar una ruta

Este ejemplo reemplaza la ruta especificada en la tabla de rutas especificada. La nueva ruta coincide con el CIDR especificado y envía el tráfico a la puerta de enlace privada virtual especificada. Si el comando se ejecuta correctamente, no se muestra ningún resultado.

Comando:

```
aws ec2 replace-route --route-table-id rtb-22574640 --destination-cidr-block  
10.0.0.0/16 --gateway-id vgw-9a4cacf3
```

- Para obtener más información sobre la API, consulte [ReplaceRoute](#) la Referencia de AWS CLI comandos.

### PowerShell

#### Herramientas para PowerShell

Ejemplo 1: Este ejemplo reemplaza la ruta especificada por la tabla de rutas especificada. La nueva ruta envía el tráfico especificado a la puerta de enlace privada virtual especificada.

```
Set-EC2Route -RouteTableId rtb-1a2b3c4d -DestinationCidrBlock 10.0.0.0/24 -  
GatewayId vgw-1a2b3c4d
```

- Para obtener más información sobre la API, consulte [ReplaceRoute](#) la referencia de AWS Tools for PowerShell cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **ReplaceRouteTableAssociation** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `ReplaceRouteTableAssociation`.

### CLI

#### AWS CLI

Para reemplazar la tabla de rutas asociada a una subred

En este ejemplo, se asocia la tabla de enrutamiento especificada a la subred de la asociación de tabla de enrutamiento especificada.

Comando:

```
aws ec2 replace-route-table-association --association-id rtbassoc-781d0d1a --  
route-table-id rtb-22574640
```

Salida:

```
{  
  "NewAssociationId": "rtbassoc-3a1f0f58"  
}
```

- Para obtener más información sobre la API, consulte [ReplaceRouteTableAssociation](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: Este ejemplo asocia la tabla de rutas especificada a la subred de la asociación de tabla de rutas especificada.

```
Set-EC2RouteTableAssociation -RouteTableId rtb-1a2b3c4d -AssociationId  
rtbassoc-12345678
```

Salida:

```
rtbassoc-87654321
```

- Para obtener más información sobre la API, consulte la referencia [ReplaceRouteTableAssociation](#) del AWS Tools for PowerShell cmdlet.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **ReportInstanceStatus** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `ReportInstanceStatus`.

### CLI

#### AWS CLI

Para informar sobre el estado de una instancia

Este comando de ejemplo informa sobre el estado de la instancia especificada.

Comando:

```
aws ec2 report-instance-status --instances i-1234567890abcdef0 --status impaired  
--reason-codes unresponsive
```

- Para obtener más información sobre la API, consulta [ReportInstanceStatus](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: Este ejemplo informa sobre el estado de la instancia especificada.

```
Send-EC2InstanceStatus -Instance i-12345678 -Status impaired -ReasonCode unresponsive
```

- Para obtener más información sobre la API, consulte [ReportInstanceStatus](#) la referencia del AWS Tools for PowerShell cmdlet.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **RequestSpotFleet** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar RequestSpotFleet.

### CLI

#### AWS CLI

Para solicitar una flota de Spot en la subred con el precio más bajo

Este comando de ejemplo crea una solicitud de flota puntual con dos especificaciones de lanzamiento que solo difieren según la subred. La flota de Spot lanza las instancias en la subred especificada con el precio más bajo. Si las instancias se lanzan en una VPC predeterminada, reciben una dirección IP pública de forma predeterminada. Si las instancias se lanzan en una VPC no predeterminada, no reciben una dirección IP pública de forma predeterminada.

Tenga en cuenta que no puede especificar subredes diferentes de la misma zona de disponibilidad en una solicitud de flota puntual.

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"
      }
    }
  ]
}
```

Salida:

```
{
  "SpotFleetRequestId": "sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE"
}
```

Para solicitar una flota puntual en la zona de disponibilidad con el precio más bajo

Este comando de ejemplo crea una solicitud de flota puntual con dos especificaciones de lanzamiento que solo difieren según la zona de disponibilidad. La flota de Spot lanza las instancias en la zona de disponibilidad especificada con el precio más bajo. Si su cuenta solo admite EC2-VPC, Amazon EC2 lanza las instancias puntuales en la subred predeterminada de la zona de disponibilidad. Si su cuenta admite EC2-Classic, Amazon EC2 lanza las instancias en EC2-Classic en la zona de disponibilidad.

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 lanzar instancias puntuales en una subred y asignarles direcciones IP públicas

Este comando de ejemplo asigna direcciones públicas a las instancias lanzadas en una VPC no predeterminada. Tenga en cuenta que, al especificar una interfaz de red, debe incluir el ID de subred y el ID del grupo de seguridad mediante la interfaz de red.

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 una flota puntual mediante la estrategia de asignación diversificada

Este comando de ejemplo crea una solicitud de flota puntual que lanza 30 instancias mediante la estrategia de asignación diversificada. Las especificaciones de lanzamiento varían según el tipo de instancia. La flota de Spot distribuye las instancias entre las especificaciones de lanzamiento, de forma que hay 10 instancias 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 obtener más información, consulte [Spot Fleet Requests](#) en la Guía del usuario de Amazon Elastic Compute Cloud.

- Para obtener más información sobre la API, consulte [RequestSpotFleet](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: Este ejemplo crea una solicitud de flota puntual en la zona de disponibilidad con el precio más bajo para el tipo de instancia especificado. Si su cuenta solo admite EC2-VPC, la flota puntual lanza las instancias en la zona de disponibilidad de menor precio que tenga una subred predeterminada. Si su cuenta es compatible con EC2-Classic, la flota de Spot lanza las instancias en EC2-Classic en la zona de disponibilidad con el precio más bajo. Tenga en cuenta que el precio que pague no superará el precio puntual especificado para la solicitud.

```
$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 obtener más información sobre la API, consulte [RequestSpotFleet](#) la referencia de AWS Tools for PowerShell cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **RequestSpotInstances** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar RequestSpotInstances.

### CLI

#### AWS CLI

Para solicitar instancias puntuales

Este comando de ejemplo crea una solicitud única de instancia puntual para cinco instancias en la zona de disponibilidad especificada. Si su cuenta solo admite EC2-VPC, Amazon EC2 lanza las instancias en la subred predeterminada de la zona de disponibilidad especificada. Si su cuenta admite EC2-Classic, Amazon EC2 lanza las instancias en EC2-Classic en la zona de disponibilidad especificada.

Comando:

```
aws ec2 request-spot-instances --spot-price "0.03" --instance-count 5 --type
"one-time" --launch-specification file://specification.json
```

Especificación.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"
  }
}
```

```
}

```

**Salida:**

```
{
  "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"
    },
    ...
  ]
}
```

Este comando de ejemplo crea una solicitud única de instancia puntual para cinco instancias de la subred especificada. Amazon EC2 inicia las instancias en la subred especificada. Si la VPC no es una VPC predeterminada, las instancias no reciben una dirección IP pública de forma predeterminada.

Comando:

```
aws ec2 request-spot-instances --spot-price "0.050" --instance-count 5 --type
"one-time" --launch-specification file://specification.json
```

Especificación.json:

```
{
  "ImageId": "ami-1a2b3c4d",
  "SecurityGroupIds": [ "sg-1a2b3c4d" ],
  "InstanceType": "m3.medium",
  "SubnetId": "subnet-1a2b3c4d",
  "IamInstanceProfile": {
    "Arn": "arn:aws:iam::123456789012:instance-profile/my-iam-role"
  }
}
```

Salida:

```
{
  "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"
},
...
]
}

```

En este ejemplo, se asigna una dirección IP pública a las instancias puntuales que se lanzan en una VPC no predeterminada. Tenga en cuenta que, al especificar una interfaz de red, debe incluir el ID de subred y el ID del grupo de seguridad mediante la interfaz de red.

Comando:

```
aws ec2 request-spot-instances --spot-price "0.050" --instance-count 1 --type
"one-time" --launch-specification file://specification.json
```

Especificación.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 obtener más información sobre la API, consulte la Referencia de [RequestSpotInstances](#) comandos AWS CLI .

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: en este ejemplo se solicita una instancia de spot única en la subred especificada. Tenga en cuenta que el grupo de seguridad debe crearse para la VPC que contiene la subred especificada y debe especificarse mediante un ID mediante la interfaz de red. Al especificar una interfaz de red, debe incluir el ID de subred mediante la interfaz de red.

```

$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

```

### Salida:

```

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 obtener más información sobre la API, consulte la Referencia [RequestSpotInstances](#) de AWS Tools for PowerShell cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **ResetImageAttribute** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `ResetImageAttribute`.

### CLI

#### AWS CLI

Para restablecer el atributo `launchPermission`

En este ejemplo, se restablece el `launchPermission` atributo de la AMI especificada a su valor predeterminado. De forma predeterminada, las AMI son privadas. Si el comando se ejecuta correctamente, no se muestra ningún resultado.

Comando:

```
aws ec2 reset-image-attribute --image-id ami-5731123e --attribute
launchPermission
```

- Para obtener información sobre la API, consulte [ResetImageAttribute](#) la Referencia de AWS CLI comandos.

### PowerShell

#### Herramientas para PowerShell

Ejemplo 1: En este ejemplo se restablece el valor predeterminado del atributo 'launchPermission'. De forma predeterminada, las AMI son privadas.

```
Reset-EC2ImageAttribute -ImageId ami-12345678 -Attribute launchPermission
```

- Para obtener información sobre la API, consulte [ResetImageAttribute](#) la referencia de AWS Tools for PowerShell cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **ResetInstanceAttribute** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `ResetInstanceAttribute`.

### CLI

#### AWS CLI

Para restablecer el `sourceDestCheck` atributo

En este ejemplo se restablece el `sourceDestCheck` atributo de la instancia especificada. La instancia debe estar en una VPC. Si el comando se ejecuta correctamente, no se muestra ningún resultado.

Comando:

```
aws ec2 reset-instance-attribute --instance-id i-1234567890abcdef0 --attribute sourceDestCheck
```

Para restablecer el atributo del núcleo

En este ejemplo se restablece el `kernel` atributo de la instancia especificada. La instancia debe tener el estado `stopped`. Si el comando se ejecuta correctamente, no se muestra ningún resultado.

Comando:

```
aws ec2 reset-instance-attribute --instance-id i-1234567890abcdef0 --attribute kernel
```

Para restablecer el atributo `ramdisk`

En este ejemplo se restablece el `ramdisk` atributo de la instancia especificada. La instancia debe tener el estado `stopped`. Si el comando se ejecuta correctamente, no se muestra ningún resultado.

Comando:

```
aws ec2 reset-instance-attribute --instance-id i-1234567890abcdef0 --attribute
ramdisk
```

- Para obtener más información sobre la API, consulte [ResetInstanceAttribute](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: en este ejemplo se restablece el atributo `sriovNetSupport` de la instancia especificada.

```
Reset-EC2InstanceAttribute -InstanceId i-12345678 -Attribute sriovNetSupport
```

Ejemplo 2: en este ejemplo se restablece el atributo `EBSOptimized` de la instancia especificada.

```
Reset-EC2InstanceAttribute -InstanceId i-12345678 -Attribute ebsOptimized
```

Ejemplo 3: en este ejemplo se restablece el atributo `sourceDestCheck` de la instancia especificada.

```
Reset-EC2InstanceAttribute -InstanceId i-12345678 -Attribute sourceDestCheck
```

Ejemplo 4: en este ejemplo se restablece el atributo `disableApiTermination` de la instancia especificada.

```
Reset-EC2InstanceAttribute -InstanceId i-12345678 -Attribute
disableApiTermination
```

Ejemplo 5: en este ejemplo se restablece el atributo `instanceInitiatedShutdownBehavior` de la instancia especificada.

```
Reset-EC2InstanceAttribute -InstanceId i-12345678 -Attribute
instanceInitiatedShutdownBehavior
```

- Para obtener más información sobre la API, consulte la referencia del [ResetInstanceAttribute](#) cmdlet AWS Tools for PowerShell .

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **ResetNetworkInterfaceAttribute** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `ResetNetworkInterfaceAttribute`.

### CLI

#### AWS CLI

Para restablecer un atributo de la interfaz de red

El siguiente `reset-network-interface-attribute` ejemplo restablece el valor del atributo de comprobación de origen/destino en. `true`

```
aws ec2 reset-network-interface-attribute \
  --network-interface-id eni-686ea200 \
  --source-dest-check
```

Este comando no genera ninguna salida.

- Para obtener más información sobre la API, consulte la Referencia de [ResetNetworkInterfaceAttribute](#) comandos AWS CLI .

### PowerShell

#### Herramientas para PowerShell

Ejemplo 1: En este ejemplo se restablece la comprobación de origen y destino de la interfaz de red especificada.

```
Reset-EC2NetworkInterfaceAttribute -NetworkInterfaceId eni-1a2b3c4d -  
SourceDestCheck
```

- Para obtener más información sobre la API, consulte la referencia de [ResetNetworkInterfaceAttribute](#) cmdlets AWS Tools for PowerShell .

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **ResetSnapshotAttribute** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `ResetSnapshotAttribute`.

### CLI

#### AWS CLI

Para restablecer un atributo de instantánea

En este ejemplo, se restablecen los permisos de creación de volúmenes para la instantánea `snap-1234567890abcdef0`. Si el comando se ejecuta correctamente, no se muestra ningún resultado.

Comando:

```
aws ec2 reset-snapshot-attribute --snapshot-id snap-1234567890abcdef0 --attribute  
createVolumePermission
```

- Para obtener más información sobre la API, consulte [ResetSnapshotAttribute](#) la Referencia de AWS CLI comandos.

### PowerShell

#### Herramientas para PowerShell

Ejemplo 1: en este ejemplo se restablece el atributo especificado de la instantánea especificada.

```
Reset-EC2SnapshotAttribute -SnapshotId snap-12345678 -Attribute  
CreateVolumePermission
```

- Para obtener más información sobre la API, consulte la referencia [ResetSnapshotAttribute](#) del AWS Tools for PowerShell cmdlet.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **RevokeSecurityGroupEgress** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `RevokeSecurityGroupEgress`.

### CLI

#### AWS CLI

Ejemplo 1: Para eliminar la regla que permite el tráfico saliente a un rango de direcciones específico

El siguiente comando de `revoke-security-group-egress` ejemplo elimina la regla que concede acceso a los rangos de direcciones especificados en el puerto TCP 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 no genera ninguna salida.

Para obtener más información, consulte [Grupos de seguridad](#) en la Guía del usuario de Amazon EC2.

Ejemplo 2: Eliminar la regla que permite el tráfico saliente a un grupo de seguridad específico

El siguiente comando de `revoke-security-group-egress` ejemplo elimina la regla que concede acceso al grupo de seguridad especificado en el puerto TCP 80.

```
aws ec2 revoke-security-group-egress \  
  --group-id sg-026c12253ce15eff7 \  
  --ip-permissions
```

```
--ip-permissions '[{"IpProtocol": "tcp", "FromPort": 443, "ToPort": 443, "UserIdGroupPairs": [{"GroupId": "sg-06df23a01ff2df86d"}]]'
```

Este comando no genera ninguna salida.

Para obtener más información, consulte [Grupos de seguridad](#) en la Guía del usuario de Amazon EC2.

- Para obtener más información sobre la API, consulte [RevokeSecurityGroupEgress](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: este ejemplo elimina la regla del grupo de seguridad especificado para EC2-VPC. Esto revoca el acceso al rango de direcciones IP especificado en el puerto TCP 80. La sintaxis utilizada en este ejemplo requiere la PowerShell versión 3 o superior.

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

Ejemplo 2: Con la PowerShell versión 2, debe usar `New-Object` para crear el `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
```

Ejemplo 3: Este ejemplo revoca el acceso al grupo de seguridad de origen especificado en el puerto TCP 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 obtener más información sobre la API, consulte la referencia [RevokeSecurityGroupEgress](#) de AWS Tools for PowerShell cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **RevokeSecurityGroupIngress** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `RevokeSecurityGroupIngress`.

### CLI

#### AWS CLI

Ejemplo 1: Para eliminar una regla de un grupo de seguridad

En el siguiente `revoke-security-group-ingress` ejemplo, se quita el acceso al puerto TCP 22 al rango de `203.0.113.0/24` direcciones del grupo de seguridad especificado para una VPC predeterminada.

```
aws ec2 revoke-security-group-ingress \  
  --group-name mySecurityGroup \  
  --protocol tcp \  
  --port 22 \  
  --cidr 203.0.113.0/24
```

Este comando no produce ningún resultado si se ejecuta correctamente.

Para obtener más información, consulte [Grupos de seguridad](#) en la Guía del usuario de Amazon EC2.

Ejemplo 2: Para eliminar una regla mediante el conjunto de permisos de IP

En el siguiente `revoke-security-group-ingress` ejemplo, se utiliza el `ip-permissions` parámetro para eliminar una regla de entrada que permite el mensaje ICMP Destination Unreachable: Fragmentation Needed and Don't Fragment was Set (tipo 3, código 4).

```
aws ec2 revoke-security-group-ingress \  
  --group-name mySecurityGroup \  
  --protocol icmp \  
  --port 0-255 \  
  --cidr 203.0.113.0/24 \  
  --ip-permissions ip-permissions=3,4
```

```
--group-id sg-026c12253ce15eff7 \  
--ip-permissions  
IpProtocol=icmp,FromPort=3,ToPort=4,IpRanges=[{CidrIp=0.0.0.0/0}]
```

Este comando no produce ningún resultado si se ejecuta correctamente.

Para obtener más información, consulte [Grupos de seguridad](#) en la Guía del usuario de Amazon EC2.

- Para obtener más información sobre la API, consulte [RevokeSecurityGroupIngress](#) la Referencia de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: este ejemplo revoca el acceso al puerto TCP 22 desde el rango de direcciones especificado para el grupo de seguridad especificado para EC2-VPC. Tenga en cuenta que debe identificar los grupos de seguridad para EC2-VPC mediante el ID del grupo de seguridad, no el nombre del grupo de seguridad. La sintaxis utilizada en este ejemplo requiere la PowerShell versión 3 o superior.

```
$ip = @{ IpProtocol="tcp"; FromPort="22"; ToPort="22";  
IpRanges="203.0.113.0/24" }  
Revoke-EC2SecurityGroupIngress -GroupId sg-12345678 -IpPermission $ip
```

Ejemplo 2: Con la PowerShell versión 2, debe usar New-Object para crear el 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
```

Ejemplo 3: Este ejemplo revoca el acceso al puerto TCP 22 desde el rango de direcciones especificado para el grupo de seguridad especificado para EC2-Classic. La sintaxis utilizada en este ejemplo requiere la PowerShell versión 3 o superior.

```
$ip = @{ IpProtocol="tcp"; FromPort="22"; ToPort="22";  
  IpRanges="203.0.113.0/24" }  
  
Revoke-EC2SecurityGroupIngress -GroupName "my-security-group" -IpPermission $ip
```

Ejemplo 4: Con la PowerShell versión 2, debe usar `New-Object` para crear el `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 obtener más información sobre la API, consulte la referencia de [RevokeSecurityGroupIngress AWS Tools for PowerShell](#) cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **RunInstances** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `RunInstances`.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en contexto en el siguiente ejemplo de código:

- [Comience a utilizar instancias](#)

## .NET

### AWS SDK for .NET

#### Note

Hay más información al respecto en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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)
{
    var request = new RunInstancesRequest
    {
        ImageId = imageId,
        InstanceType = instanceType,
        KeyName = keyName,
        MinCount = 1,
        MaxCount = 1,
        SecurityGroupIds = new List<string> { groupId }
    };
    var response = await _amazonEC2.RunInstancesAsync(request);
    return response.Reservation.Instances[0].InstanceId;
}
```

- Para obtener más información sobre la API, consulta [RunInstances](#) la Referencia AWS SDK for .NET de la API.

## Bash

### AWS CLI con el script Bash

#### Note

Hay más información. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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
}

```

Las funciones de utilidad utilizadas en este ejemplo.

```

#####
# 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 obtener más información sobre la API, consulta [RunInstances](#) la Referencia de AWS CLI comandos.



## C++

## SDK para C++

 Note

Hay más información al respecto en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

```
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();
```

- Para obtener más información sobre la API, consulta [RunInstances](#) la Referencia AWS SDK for C++ de la API.

## CLI

## AWS CLI

## Ejemplo 1: Lanzar una instancia en una subred predeterminada

En el siguiente ejemplo de `run-instances`, se lanza una única instancia de tipo `t2.micro` en la subred predeterminada de la región actual y se la asocia a la subred predeterminada de la VPC predeterminada de la región. El par de claves es opcional si no tiene pensado conectarse a la instancia mediante SSH (Linux) o RDP (Windows).

```
aws ec2 run-instances \  
  --image-id ami-0abcdef1234567890 \  
  --instance-type t2.micro \  
  --key-name MyKeyPair
```

## Salida:

```
{  
  "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"
}

```

**Ejemplo 2: Lanzar una instancia en una subred no predeterminada y agregar una dirección IP pública**

En el siguiente ejemplo de `run-instances`, se solicita una dirección IP pública para una instancia que se va a lanzar en una subred no predeterminada. La instancia se asocia a los grupos de seguridad especificados.

```
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 ver un ejemplo del resultado de `run-instances`, consulte el ejemplo 1.

### Ejemplo 3: Lanzar una instancia con volúmenes adicionales

En el siguiente ejemplo de `run-instances`, se usa una asignación de dispositivos de bloques, especificada en `mapping.json`, para asociar volúmenes adicionales en el momento del lanzamiento. Una asignación de dispositivos de bloques puede especificar volúmenes de EBS, volúmenes de almacén de instancias o tanto volúmenes de EBS como volúmenes de almacén de instancias.

```
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
```

Contenido de `mapping.json`. En este ejemplo, se le agrega a `/dev/sdh` un volumen de EBS vacío con un tamaño de 100 GiB.

```
[  
  {  
    "DeviceName": "/dev/sdh",  
    "Ebs": {  
      "VolumeSize": 100  
    }  
  }  
]
```

Contenido de `mapping.json`. En este ejemplo, se le agrega a `ephemeral1` un volumen de almacén de instancias.

```
[
```

```
{
  "DeviceName": "/dev/sdc",
  "VirtualName": "ephemeral1"
}
]
```

Para ver un ejemplo del resultado de `run-instances`, consulte el ejemplo 1.

Para obtener más información acerca de la asignación de dispositivos de bloques, consulte [Asignación de dispositivos de bloques](#) en la Guía del usuario de Amazon EC2.

#### Ejemplo 4: Lanzar una instancia y agregar etiquetas al crearla

En el siguiente ejemplo de `run-instances`, se agrega una etiqueta con una clave de `webserver` y un valor de `production` a la instancia. El comando también aplica una etiqueta con una clave de `cost-center` y un valor de `cc123` a cualquier volumen de EBS que se cree (en este caso, el volumen raíz).

```
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 ver un ejemplo del resultado de `run-instances`, consulte el ejemplo 1.

#### Ejemplo 5: Lanzar una instancia con datos de usuario

En el siguiente ejemplo de `run-instances`, se transfieren los datos del usuario a un archivo denominado `my_script.txt` que contiene un script de configuración para la instancia. El script se ejecuta en el momento del lanzamiento.

```
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 ver un ejemplo del resultado de `run-instances`, consulte el ejemplo 1.

Para obtener más información, consulte [Trabajo con datos de usuario en instancias](#) en la Guía del usuario de Amazon EC2.

### Ejemplo 6: Lanzar una instancia de rendimiento ampliable

En el siguiente ejemplo de `run-instances`, se lanza una instancia `t2.micro` con la opción de crédito `unlimited`. Al lanzar una instancia T2, si no especifica `--credit-specification`, la opción de crédito predeterminada es `standard`. Al lanzar una instancia T3, la opción de crédito predeterminada es `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 ver un ejemplo del resultado de `run-instances`, consulte el ejemplo 1.

Para obtener más información sobre las instancias de rendimiento ampliable, consulte [Instancias de rendimiento ampliable](#) en la Guía del usuario de Amazon EC2.

- Para obtener más información sobre la API, consulta [RunInstances](#) la Referencia de AWS CLI comandos.

## Java

### SDK para Java 2.x

#### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.ec2.Ec2Client;
import software.amazon.awssdk.services.ec2.model.InstanceType;
import software.amazon.awssdk.services.ec2.model.RunInstancesRequest;
import software.amazon.awssdk.services.ec2.model.RunInstancesResponse;
import software.amazon.awssdk.services.ec2.model.Tag;
import software.amazon.awssdk.services.ec2.model.CreateTagsRequest;
import software.amazon.awssdk.services.ec2.model.Ec2Exception;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
 *
 * This code example requires an AMI value. You can learn more about this value
 * by reading this documentation topic:
 *
 * https://docs.aws.amazon.com/AWSEC2/latest/WindowsGuide/AMIs.html
 */
public class CreateInstance {
    public static void main(String[] args) {
        final String usage = ""

            Usage:
                <name> <amiId>

            Where:
                name - An instance name value that you can obtain from the AWS
                Console (for example, ami-xxxxxx5c8b987b1a0).\s
                amiId - An Amazon Machine Image (AMI) value that you can
                obtain from the AWS Console (for example, i-xxxxxx2734106d0ab).\s
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String name = args[0];
```



```
String amiId = args[1];
Region region = Region.US_EAST_1;
Ec2Client ec2 = Ec2Client.builder()
    .region(region)
    .build();

String instanceId = createEC2Instance(ec2, name, amiId);
System.out.println("The Amazon EC2 Instance ID is " + instanceId);
ec2.close();
}

public static String createEC2Instance(Ec2Client ec2, String name, String
amiId) {
    RunInstancesRequest runRequest = RunInstancesRequest.builder()
        .imageId(amiId)
        .instanceType(InstanceType.T1_MICRO)
        .maxCount(1)
        .minCount(1)
        .build();

    // Use a waiter to wait until the instance is running.
    System.out.println("Going to start an EC2 instance using a waiter");
    RunInstancesResponse response = ec2.runInstances(runRequest);
    String instanceIdVal = response.instances().get(0).instanceId();
    ec2.waiter().waitUntilInstanceRunning(r -> r.instanceIds(instanceIdVal));
    Tag tag = Tag.builder()
        .key("Name")
        .value(name)
        .build();

    CreateTagsRequest tagRequest = CreateTagsRequest.builder()
        .resources(instanceIdVal)
        .tags(tag)
        .build();

    try {
        ec2.createTags(tagRequest);
        System.out.printf("Successfully started EC2 Instance %s based on AMI
%s", instanceIdVal, amiId);
        return instanceIdVal;
    } catch (Ec2Exception e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
}
```

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

- Para obtener más información sobre la API, consulta [RunInstances](#) la Referencia AWS SDK for Java 2.x de la API.

## JavaScript

### SDK para JavaScript (v3)

#### Note

Hay más información. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

```
import { RunInstancesCommand } from "@aws-sdk/client-ec2";  
  
import { client } from "../libs/client.js";  
  
// Create a new EC2 instance.  
export const main = async () => {  
  const command = new RunInstancesCommand({  
    // Your key pair name.  
    KeyName: "KEY_PAIR_NAME",  
    // Your security group.  
    SecurityGroupIds: ["SECURITY_GROUP_ID"],  
    // An x86_64 compatible image.  
    ImageId: "ami-0001a0d1a04bfcc30",  
    // An x86_64 compatible free-tier instance type.  
    InstanceType: "t1.micro",  
    // Ensure only 1 instance launches.  
    MinCount: 1,  
    MaxCount: 1,  
  });  
  
  try {
```

```
const response = await client.send(command);
console.log(response);
} catch (err) {
  console.error(err);
}
};
```

- Para obtener más información sobre la API, consulta [RunInstances](#) la Referencia AWS SDK for JavaScript de la API.

## Kotlin

### SDK para Kotlin

#### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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 obtener más información sobre la API, consulta [RunInstances](#) la referencia sobre el AWS SDK para la API de Kotlin.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: este ejemplo lanza una sola instancia de la AMI especificada en EC2-Classic o en una VPC predeterminada.

```
New-EC2Instance -ImageId ami-12345678 -MinCount 1 -MaxCount 1 -InstanceType
m3.medium -KeyName my-key-pair -SecurityGroup my-security-group
```

Ejemplo 2: este ejemplo lanza una sola instancia de la AMI especificada en una VPC.

```
New-EC2Instance -ImageId ami-12345678 -MinCount 1 -MaxCount 1 -SubnetId
subnet-12345678 -InstanceType t2.micro -KeyName my-key-pair -SecurityGroupId
sg-12345678
```

Ejemplo 3: Para añadir un volumen de EBS o un volumen de almacén de instancias, defina una asignación de dispositivos de bloques y agréguela al comando. En este ejemplo, se agrega un volumen de almacén de instancias.

```
$bdm = New-Object Amazon.EC2.Model.BlockDeviceMapping
$bdm.VirtualName = "ephemeral0"
$bdm.DeviceName = "/dev/sdf"
```

```
New-EC2Instance -ImageId ami-12345678 -BlockDeviceMapping $bdm ...
```

Ejemplo 4: Para especificar una de las AMI de Windows actuales, obtenga su ID de AMI mediante `Get-EC2ImageByName`. En este ejemplo, se lanza una instancia desde la AMI base actual para Windows Server 2016.

```
$ami = Get-EC2ImageByName WINDOWS_2016_BASE  
  
New-EC2Instance -ImageId $ami.ImageId ...
```

Ejemplo 5: lanza una instancia en el entorno 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
```

Ejemplo 6: Esta solicitud lanza dos instancias y aplica a las instancias una etiqueta con una clave del servidor web y un valor de producción. La solicitud también aplica una etiqueta con la clave `cost-center` y un valor de `cc123` a los volúmenes que se crean (en este caso, el volumen raíz de cada instancia).

```
$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 obtener más información sobre la API, consulte la referencia de [RunInstances AWS Tools for PowerShell](#) cmdlets.

## Python

### SDK para Python (Boto3)

#### Note

Hay más información al respecto. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

```
class InstanceWrapper:
    """Encapsulates Amazon Elastic Compute Cloud (Amazon EC2) instance
    actions."""

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

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

    def create(self, image, instance_type, key_pair, security_groups=None):
        """
        Creates a new EC2 instance. The instance starts immediately after
        it is created.

        The instance is created in the default VPC of the current account.

        :param image: A Boto3 Image object that represents an Amazon Machine
        Image (AMI)
```

```

        that defines attributes of the instance that is created.
The AMI
        defines things like the kind of operating system and the
type of
        storage used by the instance.
        :param instance_type: The type of instance to create, such as 't2.micro'.
        The instance type defines things like the number of
CPUs and
        the amount of memory.
        :param key_pair: A Boto3 KeyPair or KeyPairInfo object that represents
the key
        pair that is used to secure connections to the instance.
        :param security_groups: A list of Boto3 SecurityGroup objects that
represents the
        security groups that are used to grant access to
the
        instance. When no security groups are specified,
the
        default security group of the VPC is used.
        :return: A Boto3 Instance object that represents the newly created
instance.
    """
    try:
        instance_params = {
            "ImageId": image.id,
            "InstanceType": instance_type,
            "KeyName": key_pair.name,
        }
        if security_groups is not None:
            instance_params["SecurityGroupIds"] = [sg.id for sg in
security_groups]
        self.instance = self.ec2_resource.create_instances(
            **instance_params, MinCount=1, MaxCount=1
        )[0]
        self.instance.wait_until_running()
    except ClientError as err:
        logging.error(
            "Couldn't create instance with image %s, instance type %s, and
key %s. "
            "Here's why: %s: %s",
            image.id,
            instance_type,
            key_pair.name,
            err.response["Error"]["Code"],

```

```

        err.response["Error"]["Message"],
    )
    raise
else:
    return self.instance

```

- Para obtener más información sobre la API, consulta [RunInstances](#) la AWS Referencia de API de SDK for Python (Boto3).

## SAP ABAP

### SDK de SAP ABAP

#### Note

Hay más información al respecto. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

```

" Create tags for resource created during instance launch. "
DATA lt_tag specifications TYPE /aws1/
cl_ec2tag specifications=>tt_tag specification list.
DATA ls_tag specifications LIKE LINE OF lt_tag specifications.
ls_tag specifications = NEW /aws1/cl_ec2tag specification(
    iv_resource type = 'instance'
    it_tags = VALUE /aws1/cl_ec2tag=>tt_tag list(
        ( NEW /aws1/cl_ec2tag( iv_key = 'Name' iv_value = iv_tag_value ) )
    )
).
APPEND ls_tag specifications TO lt_tag specifications.

TRY.
    " Create/launch Amazon Elastic Compute Cloud (Amazon EC2) instance. "
    oo_result = lo_ec2->run instances( " oo_result
is returned for testing purposes. "
    iv_image id = iv_ami_id
    iv_instance type = 't2.micro'
    iv_max count = 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'.
    ENTRY.

```

- Para obtener más información sobre la API, consulte [RunInstances](#) la referencia sobre la API ABAP del AWS SDK para SAP.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte [Cree recursos de Amazon EC2 mediante un SDK AWS](#). En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **RunScheduledInstances** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar RunScheduledInstances.

### CLI

#### AWS CLI

Para lanzar una instancia programada

En este ejemplo, se lanza la instancia programada especificada en una 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

```

Launch-Specification.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"
}
}
```

Salida:

```
{
  "InstanceIdSet": [
    "i-1234567890abcdef0"
  ]
}
```

En este ejemplo, se lanza la instancia programada especificada en 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
```

Launch-Specification.json:

```
{
  "ImageId": "ami-12345678",
  "KeyName": "my-key-pair",
  "SecurityGroupIds": ["sg-12345678"],
  "InstanceType": "c4.large",
  "Placement": {
    "AvailabilityZone": "us-west-2b"
  }
  "IamInstanceProfile": {
    "Name": "my-iam-role"
  }
}
```

```
}
```

Salida:

```
{
  "InstanceIdSet": [
    "i-1234567890abcdef0"
  ]
}
```

- Para obtener más información sobre la API, consulte la Referencia de comandos. [RunScheduledInstances](#) AWS CLI

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: en este ejemplo se lanza la instancia programada 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 obtener más información sobre la API, consulte [RunScheduledInstances](#) la referencia del AWS Tools for PowerShell cmdlet.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **StartInstances** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `StartInstances`.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en contexto en el siguiente ejemplo de código:

- [Comience a utilizar instancias](#)

## .NET

### AWS SDK for .NET

#### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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)
{
    var request = new StartInstancesRequest
    {
        InstanceIds = new List<string> { ec2InstanceId },
    };

    var response = await _amazonEC2.StartInstancesAsync(request);

    if (response.StartingInstances.Count > 0)
    {
        var instances = response.StartingInstances;
        instances.ForEach(i =>
        {
            Console.WriteLine($"Successfully started the EC2 instance with
instance ID: {i.InstanceId}.");
        });
    }
}
```

- Para obtener más información sobre la API, consulta [StartInstances](#) la Referencia AWS SDK for .NET de la API.

## Bash

## AWS CLI con el script Bash

 Note

Hay más información. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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)."
        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

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
}

```

Las funciones de utilidad utilizadas en este ejemplo.

```

#####
# 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 obtener más información sobre la API, consulta [StartInstances](#) la Referencia de AWS CLI comandos.

## C++

## SDK para C++

 Note

Hay más información al respecto en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

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

Aws::EC2::Model::StartInstancesRequest start_request;
start_request.AddInstanceIds(instanceId);
start_request.SetDryRun(true);

auto dry_run_outcome = ec2Client.StartInstances(start_request);
if (dry_run_outcome.IsSuccess()) {
    std::cerr
        << "Failed dry run to start instance. A dry run should trigger an
error."
        << std::endl;
    return false;
}
else if (dry_run_outcome.GetError().GetErrorType() !=
    Aws::EC2::EC2Errors::DRY_RUN_OPERATION) {
    std::cout << "Failed dry run to start instance " << instanceId << ": "
        << dry_run_outcome.GetError().GetMessage() << std::endl;
    return false;
}

start_request.SetDryRun(false);
auto start_instancesOutcome = ec2Client.StartInstances(start_request);

if (!start_instancesOutcome.IsSuccess()) {
    std::cout << "Failed to start instance " << instanceId << ": " <<
        start_instancesOutcome.GetError().GetMessage() << std::endl;
}
else {
    std::cout << "Successfully started instance " << instanceId <<
        std::endl;
}
}
```



- Para obtener más información sobre la API, consulta [StartInstances](#) la Referencia AWS SDK for C++ de la API.

## CLI

### AWS CLI

Iniciar una instancia de Amazon EC2

En este ejemplo, se inicia la instancia respaldada por Amazon EBS.

Comando:

```
aws ec2 start-instances --instance-ids i-1234567890abcdef0
```

Salida:

```
{
  "StartingInstances": [
    {
      "InstanceId": "i-1234567890abcdef0",
      "CurrentState": {
        "Code": 0,
        "Name": "pending"
      },
      "PreviousState": {
        "Code": 80,
        "Name": "stopped"
      }
    }
  ]
}
```

Para obtener más información, consulte Detener e iniciar la instancia en la Guía del usuario de Amazon Elastic Compute Cloud.

- Para obtener más información sobre la API, consulta [StartInstances](#) la Referencia de AWS CLI comandos.

## Java

## SDK para Java 2.x

 Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

```
public static void startInstance(Ec2Client ec2, String instanceId) {
    Ec2Waiter ec2Waiter = Ec2Waiter.builder()
        .overrideConfiguration(b -> b.maxAttempts(100))
        .client(ec2)
        .build();

    StartInstancesRequest request = StartInstancesRequest.builder()
        .instanceIds(instanceId)
        .build();

    System.out.println("Use an Ec2Waiter to wait for the instance to run.
This will take a few minutes.");
    ec2.startInstances(request);
    DescribeInstancesRequest instanceRequest =
DescribeInstancesRequest.builder()
        .instanceIds(instanceId)
        .build();

    WaiterResponse<DescribeInstancesResponse> waiterResponse =
ec2Waiter.waitUntilInstanceRunning(instanceRequest);
    waiterResponse.matched().response().ifPresent(System.out::println);
    System.out.println("Successfully started instance " + instanceId);
}
```

- Para obtener más información sobre la API, consulta [StartInstances](#) la Referencia AWS SDK for Java 2.x de la API.

## JavaScript

### SDK para JavaScript (v3)

#### Note

Hay más información. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

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

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

export const main = async () => {
  const command = new StartInstancesCommand({
    // Use DescribeInstancesCommand to find InstanceIds
    InstanceIds: ["INSTANCE_ID"],
  });

  try {
    const { StartingInstances } = await client.send(command);
    const instanceIdList = StartingInstances.map(
      (instance) => ` • ${instance.InstanceId}`,
    );
    console.log("Starting instances:");
    console.log(instanceIdList.join("\n"));
  } catch (err) {
    console.error(err);
  }
};
```

- Para obtener más información sobre la API, consulta [StartInstances](#) la Referencia AWS SDK for JavaScript de la API.

## Kotlin

### SDK para Kotlin

#### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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 obtener más información sobre la API, consulta [StartInstances](#) la referencia sobre el AWS SDK para la API de Kotlin.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: en este ejemplo se inicia la instancia especificada.

```
Start-EC2Instance -InstanceId i-12345678
```

Salida:

CurrentState	InstanceId	PreviousState
-----	-----	-----
Amazon.EC2.Model.InstanceState	i-12345678	Amazon.EC2.Model.InstanceState

Ejemplo 2: en este ejemplo se inician las instancias especificadas.

```
@("i-12345678", "i-76543210") | Start-EC2Instance
```

Ejemplo 3: En este ejemplo se inicia el conjunto de instancias que están detenidas actualmente. Los objetos de instancia devueltos por Get-EC2Instance se canalizan a Start-EC2Instance. La sintaxis utilizada en este ejemplo requiere la PowerShell versión 3 o superior.

```
(Get-EC2Instance -Filter @{ Name="instance-state-name";
  Values="stopped"}).Instances | Start-EC2Instance
```

Ejemplo 4: Con la PowerShell versión 2, debe usar New-Object para crear el filtro para el parámetro Filter.

```
$filter = New-Object Amazon.EC2.Model.Filter
$filter.Name = "instance-state-name"
$filter.Values = "stopped"

(Get-EC2Instance -Filter $filter).Instances | Start-EC2Instance
```

- Para obtener más información sobre la API, consulte la referencia del [StartInstances AWS Tools for PowerShell](#) cmdlet.

## Python

### SDK para Python (Boto3)

#### Note

Hay más información al respecto. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

```
class InstanceWrapper:
```

```
"""Encapsulates Amazon Elastic Compute Cloud (Amazon EC2) instance
actions."""

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

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

    def start(self):
        """
        Starts an instance and waits for it to be in a running state.

        :return: The response to the start request.
        """
        if self.instance is None:
            logger.info("No instance to start.")
            return

        try:
            response = self.instance.start()
            self.instance.wait_until_running()
        except ClientError as err:
            logger.error(
                "Couldn't start instance %s. Here's why: %s: %s",
                self.instance.id,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
```

```
return response
```

- Para obtener más información sobre la API, consulta [StartInstances](#) la AWS Referencia de API de SDK for Python (Boto3).

## Ruby

### SDK para Ruby

#### Note

Hay más información al respecto. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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."
return true
rescue StandardError => e
  puts "Error starting instance: #{e.message}"
  return false
end

# Example usage:
def run_me
  instance_id = ""
  region = ""
  # Print usage information and then stop.
  if ARGV[0] == "--help" || ARGV[0] == "-h"
    puts "Usage:  ruby ec2-ruby-example-start-instance-i-123abc.rb " \
      "INSTANCE_ID REGION "
  # Replace us-west-2 with the AWS Region you're using for Amazon EC2.
  puts "Example: ruby ec2-ruby-example-start-instance-i-123abc.rb " \
    "i-123abc us-west-2"
  exit 1
  # If no values are specified at the command prompt, use these default values.
  # Replace us-west-2 with the AWS Region you're using for Amazon EC2.
  elsif ARGV.count.zero?
    instance_id = "i-123abc"
    region = "us-west-2"
  # Otherwise, use the values as specified at the command prompt.
  else
    instance_id = ARGV[0]
    region = ARGV[1]
  end
end
```



```

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

puts "Attempting to start instance '#{instance_id}' " \
      "(this might take a few minutes)..."
unless instance_started?(ec2_client, instance_id)
  puts "Could not start instance."
end
end
end

run_me if $PROGRAM_NAME == __FILE__

```

- Para obtener más información sobre la API, consulta [StartInstances](#) la Referencia AWS SDK for Ruby de la API.

## Rust

### SDK para Rust

#### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

```

async fn start_instance(client: &Client, id: &str) -> Result<(), Error> {
  // start_instance has no unique errors to handle.
  client.start_instances().instance_ids(id).send().await?;

  println!("Waiting for instance to be running");

  let wait_for_running = client
    .wait_until_instance_running()
    .instance_ids(id)
    .wait(Duration::from_secs(60))
    .await;

  match wait_for_running {
    Ok(_) => println!("Instance is running"),
    Err(err) => match err {
      WaiterError::ExceededMaxWait(exceeded) => {

```

```

        println!(
            "Exceeded max time waiting for instance to start. Exceeded
    {}s by {}s.",
            exceeded.max_wait().as_secs(),
            (exceeded.elapsed() - exceeded.max_wait()).as_secs()
        );
        return Ok(());
    }
    _ => return Err(err.into()),
},
}

println!("Started instance.");

Ok(())
}

```

- Para obtener más información sobre la API, consulta [StartInstances](#) la referencia sobre la API de AWS SDK para Rust.

## SAP ABAP

### SDK de SAP ABAP

#### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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.

```

- Para obtener más información sobre la API, consulte [StartInstances](#) la referencia sobre la API ABAP del AWS SDK para SAP.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte [Cree recursos de Amazon EC2 mediante un SDK AWS](#). En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **StopInstances** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar StopInstances.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en contexto en el siguiente ejemplo de código:

- [Comience a utilizar instancias](#)

## .NET

### AWS SDK for .NET

#### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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)
{
    // In addition to the list of instance Ids, the
    // request can also include the following properties:
    //     Force      When true, forces the instances to
    //                 stop but you must check the integrity
    //                 of the file system. Not recommended on
    //                 Windows instances.
    //     Hibernate  When true, hibernates the instance if the
    //                 instance was enabled for hibernation when
    //                 it was launched.
    var request = new StopInstancesRequest
    {
        InstanceIds = new List<string> { ec2InstanceId },
    };

    var response = await _amazonEC2.StopInstancesAsync(request);

    if (response.StoppingInstances.Count > 0)
    {
        var instances = response.StoppingInstances;
        instances.ForEach(i =>
        {
            Console.WriteLine($"Successfully stopped the EC2 Instance " +
```

```

        $"with InstanceID: {i.InstanceId}.");
    });
}
}

```

- Para obtener más información sobre la API, consulta [StopInstances](#) la Referencia AWS SDK for .NET de la API.

## Bash

### AWS CLI con el script Bash

#### Note

Hay más información. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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
```

```
}
```

Las funciones de utilidad utilizadas en este ejemplo.

```
#####  
# 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 obtener más información sobre la API, consulta [StopInstances](#) la Referencia de AWS CLI comandos.

## C++

### SDK para C++

#### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

```
Aws::EC2::EC2Client ec2Client(clientConfiguration);
Aws::EC2::Model::StopInstancesRequest request;
request.AddInstanceIds(instanceId);
request.SetDryRun(true);

auto dry_run_outcome = ec2Client.StopInstances(request);
if (dry_run_outcome.IsSuccess()) {
    std::cerr
        << "Failed dry run to stop instance. A dry run should trigger an
error."
        << std::endl;
    return false;
}
else if (dry_run_outcome.GetError().GetErrorType() !=
    Aws::EC2::EC2Errors::DRY_RUN_OPERATION) {
    std::cout << "Failed dry run to stop instance " << instanceId << ": "
        << dry_run_outcome.GetError().GetMessage() << std::endl;
    return false;
}

request.SetDryRun(false);
auto outcome = ec2Client.StopInstances(request);
if (!outcome.IsSuccess()) {
    std::cout << "Failed to stop instance " << instanceId << ": " <<
        outcome.GetError().GetMessage() << std::endl;
}
else {
    std::cout << "Successfully stopped instance " << instanceId <<
```



```
        std::endl;
    }
```

- Para obtener más información sobre la API, consulta [StopInstances](#) la Referencia AWS SDK for C++ de la API.

## CLI

### AWS CLI

#### Ejemplo 1: Detener una instancia de Amazon EC2

En el siguiente ejemplo de `stop-instances`, se detiene la instancia respaldada por Amazon EBS especificada.

```
aws ec2 stop-instances \
  --instance-ids i-1234567890abcdef0
```

Salida:

```
{
  "StoppingInstances": [
    {
      "InstanceId": "i-1234567890abcdef0",
      "CurrentState": {
        "Code": 64,
        "Name": "stopping"
      },
      "PreviousState": {
        "Code": 16,
        "Name": "running"
      }
    }
  ]
}
```

Para obtener más información, consulte [Detener e iniciar la instancia](#) en la Guía del usuario de Amazon Elastic Compute Cloud.

#### Ejemplo 2: Hibernar una instancia de Amazon EC2

En el siguiente ejemplo de `stop-instances`, se hiberna una instancia respaldada por Amazon EBS siempre y cuando la instancia sea compatible con la hibernación y cumpla con los requisitos previos para la hibernación. Luego de que la instancia sea puesta en hibernación, la instancia se detiene.

```
aws ec2 stop-instances \  
  --instance-ids i-1234567890abcdef0 \  
  --hibernate
```

Salida:

```
{  
  "StoppingInstances": [  
    {  
      "CurrentState": {  
        "Code": 64,  
        "Name": "stopping"  
      },  
      "InstanceId": "i-1234567890abcdef0",  
      "PreviousState": {  
        "Code": 16,  
        "Name": "running"  
      }  
    }  
  ]  
}
```

Para obtener más información, consulte [Hibernar la instancia de Linux bajo demanda](#) en la Guía del usuario de Amazon Elastic Compute Cloud.

- Para obtener más información sobre la API, consulta [StopInstances](#) la Referencia de AWS CLI comandos.

## Java

## SDK para Java 2.x

 Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

```
public static void stopInstance(Ec2Client ec2, String instanceId) {
    Ec2Waiter ec2Waiter = Ec2Waiter.builder()
        .overrideConfiguration(b -> b.maxAttempts(100))
        .client(ec2)
        .build();
    StopInstancesRequest request = StopInstancesRequest.builder()
        .instanceIds(instanceId)
        .build();

    System.out.println("Use an Ec2Waiter to wait for the instance to stop.
This will take a few minutes.");
    ec2.stopInstances(request);
    DescribeInstancesRequest instanceRequest =
DescribeInstancesRequest.builder()
        .instanceIds(instanceId)
        .build();

    WaiterResponse<DescribeInstancesResponse> waiterResponse =
ec2Waiter.waitUntilInstanceStopped(instanceRequest);
    waiterResponse.matched().response().ifPresent(System.out::println);
    System.out.println("Successfully stopped instance " + instanceId);
}
```

- Para obtener más información sobre la API, consulta [StopInstances](#) la Referencia AWS SDK for Java 2.x de la API.

## JavaScript

### SDK para JavaScript (v3)

#### Note

Hay más información. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

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

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

export const main = async () => {
  const command = new StopInstancesCommand({
    // Use DescribeInstancesCommand to find InstanceIds
    InstanceIds: ["INSTANCE_ID"],
  });

  try {
    const { StoppingInstances } = await client.send(command);
    const instanceIdList = StoppingInstances.map(
      (instance) => ` • ${instance.InstanceId}`,
    );
    console.log("Stopping instances:");
    console.log(instanceIdList.join("\n"));
  } catch (err) {
    console.error(err);
  }
};
```

- Para obtener más información sobre la API, consulta [StopInstances](#) la Referencia AWS SDK for JavaScript de la API.

## Kotlin

### SDK para Kotlin

#### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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 obtener más información sobre la API, consulta [StopInstances](#) la referencia sobre el AWS SDK para la API de Kotlin.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: Este ejemplo detiene la instancia especificada.

```
Stop-EC2Instance -InstanceId i-12345678
```

Salida:

CurrentState	InstanceId	PreviousState
-----	-----	-----
Amazon.EC2.Model.InstanceState	i-12345678	Amazon.EC2.Model.InstanceState

- Para obtener más información sobre la API, consulte [StopInstances](#) la referencia del AWS Tools for PowerShell cmdlet.

## Python

### SDK para Python (Boto3)

#### Note

Hay más información al respecto. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

```
class InstanceWrapper:
    """Encapsulates Amazon Elastic Compute Cloud (Amazon EC2) instance
    actions."""

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

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

    def stop(self):
```

```
"""
Stops an instance and waits for it to be in a stopped state.

:return: The response to the stop request.
"""
if self.instance is None:
    logger.info("No instance to stop.")
    return

try:
    response = self.instance.stop()
    self.instance.wait_until_stopped()
except ClientError as err:
    logger.error(
        "Couldn't stop instance %s. Here's why: %s: %s",
        self.instance.id,
        err.response["Error"]["Code"],
        err.response["Error"]["Message"],
    )
    raise
else:
    return response
```

- Para obtener más información sobre la API, consulta [StopInstances](#) la AWS Referencia de API de SDK for Python (Boto3).

## Ruby

### SDK para Ruby

#### Note

Hay más información al respecto. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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

ec2_client.stop_instances(instance_ids: [instance_id])
ec2_client.wait_until(:instance_stopped, instance_ids: [instance_id])
puts "Instance stopped."
return true
rescue StandardError => e
  puts "Error stopping instance: #{e.message}"
  return false
end

# Example usage:
def run_me
  instance_id = ""
  region = ""
```



```
# Print usage information and then stop.
if ARGV[0] == "--help" || ARGV[0] == "-h"
  puts "Usage:  ruby ec2-ruby-example-stop-instance-i-123abc.rb " \
    "INSTANCE_ID REGION "
  # Replace us-west-2 with the AWS Region you're using for Amazon EC2.
  puts "Example: ruby ec2-ruby-example-start-instance-i-123abc.rb " \
    "i-123abc us-west-2"
  exit 1
# If no values are specified at the command prompt, use these default values.
# Replace us-west-2 with the AWS Region you're using for Amazon EC2.
elsif ARGV.count.zero?
  instance_id = "i-123abc"
  region = "us-west-2"
# Otherwise, use the values as specified at the command prompt.
else
  instance_id = ARGV[0]
  region = ARGV[1]
end

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

puts "Attempting to stop instance '#{instance_id}' " \
  "(this might take a few minutes)..."
unless instance_stopped?(ec2_client, instance_id)
  puts "Could not stop instance."
end
end

run_me if $PROGRAM_NAME == __FILE__
```

- Para obtener más información sobre la API, consulta [StopInstances](#) la Referencia AWS SDK for Ruby de la API.

## Rust

### SDK para Rust

#### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

```
async fn stop_instance(client: &Client, id: &str) -> Result<(), Error> {
    client.stop_instances().instance_ids(id).send().await?;

    println!("Stopping instance...");

    let wait = client
        .wait_until_instance_stopped()
        .instance_ids(id)
        .wait(Duration::from_secs(60))
        .await;

    match wait {
        Ok(_) => {
            println!("Stopped instance.");
        }
        Err(err) => match err {
            WaiterError::ExceededMaxWait(exceeded) => {
                println!(
                    "Exceeded max time waiting for instance to stop. Exceeded {}s
by {}s",
                    exceeded.max_wait().as_secs(),
                    (exceeded.elapsed() - exceeded.max_wait()).as_secs()
                )
            }
            _ => return Err(err.into()),
        },
    };
    Ok(())
}
```

- Para obtener más información sobre la API, consulta [StopInstances](#) la referencia sobre la API de AWS SDK para Rust.

## SAP ABAP

### SDK de SAP ABAP

#### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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.
ENDTRY.

```

- Para obtener más información sobre la API, consulte [StopInstances](#) la referencia sobre la API ABAP del AWS SDK para SAP.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte [Cree recursos de Amazon EC2 mediante un SDK AWS](#). En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **TerminateInstances** con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `TerminateInstances`.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en contexto en el siguiente ejemplo de código:

- [Comience a utilizar instancias](#)

.NET

AWS SDK for .NET

### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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)
    {
        var request = new TerminateInstancesRequest
        {
            InstanceIds = new List<string> { ec2InstanceId }
        };

        var response = await _amazonEC2.TerminateInstancesAsync(request);
        return response.TerminatingInstances;
    }

```

- Para obtener más información sobre la API, consulta [TerminateInstances](#) la Referencia AWS SDK for .NET de la API.

## Bash

### AWS CLI con el script Bash

#### Note

Hay más información. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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 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
}

```

Las funciones de utilidad utilizadas en este ejemplo.

```

#####
# 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 obtener más información sobre la API, consulta [TerminateInstances](#) la Referencia de AWS CLI comandos.

## C++

### SDK para C++

#### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

```

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;
}
```

- Para obtener más información sobre la API, consulta [TerminateInstances](#) la Referencia AWS SDK for C++ de la API.

## CLI

### AWS CLI

Terminar una instancia de Amazon EC2

En este ejemplo, se termina la instancia especificada.

Comando:

```
aws ec2 terminate-instances --instance-ids i-1234567890abcdef0
```

Salida:


```
{
  "TerminatingInstances": [
    {
      "InstanceId": "i-1234567890abcdef0",
      "CurrentState": {
        "Code": 32,
        "Name": "shutting-down"
      },
      "PreviousState": {
        "Code": 16,
        "Name": "running"
      }
    }
  ]
}
```

Para obtener más información, consulte [Uso de instancias de Amazon EC2](#) en la Guía del usuario de la interfaz de la línea de comandos de AWS .

- Para obtener más información sobre la API, consulta [TerminateInstances](#) la Referencia de AWS CLI comandos.

## Java

## SDK para Java 2.x

 Note

Hay más información al respecto en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

```
public static void terminateEC2(Ec2Client ec2, String instanceId) {
    try {
        Ec2Waiter ec2Waiter = Ec2Waiter.builder()
            .overrideConfiguration(b -> b.maxAttempts(100))
            .client(ec2)
            .build();

        TerminateInstancesRequest ti = TerminateInstancesRequest.builder()
            .instanceIds(instanceId)
            .build();

        System.out.println("Use an Ec2Waiter to wait for the instance to
        terminate. This will take a few minutes.");
        ec2.terminateInstances(ti);
        DescribeInstancesRequest instanceRequest =
        DescribeInstancesRequest.builder()
            .instanceIds(instanceId)
            .build();

        WaiterResponse<DescribeInstancesResponse> waiterResponse = ec2Waiter
            .waitUntilInstanceTerminated(instanceRequest);
        waiterResponse.matched().response().ifPresent(System.out::println);
        System.out.println("Successfully started instance " + instanceId);
        System.out.println(instanceId + " is terminated!");

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

- Para obtener más información sobre la API, consulta [TerminateInstances](#) la Referencia AWS SDK for Java 2.x de la API.

## JavaScript

### SDK para JavaScript (v3)

#### Note

Hay más información. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

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

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

export const main = async () => {
  const command = new TerminateInstancesCommand({
    InstanceIds: ["INSTANCE_ID"],
  });

  try {
    const { TerminatingInstances } = await client.send(command);
    const instanceList = TerminatingInstances.map(
      (instance) => ` • ${instance.InstanceId}`,
    );
    console.log("Terminating instances:");
    console.log(instanceList.join("\n"));
  } catch (err) {
    console.error(err);
  }
};
```

- Para obtener más información sobre la API, consulta [TerminateInstances](#) la Referencia AWS SDK for JavaScript de la API.

## Kotlin

### SDK para Kotlin

#### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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 obtener más información sobre la API, consulta [TerminateInstances](#) la referencia sobre el AWS SDK para la API de Kotlin.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: en este ejemplo se termina la instancia especificada (la instancia puede estar en ejecución o en estado «detenida»). El cmdlet solicitará una confirmación antes de continuar; utilice el modificador `-Force` para suprimir la solicitud.

```
Remove-EC2Instance -InstanceId i-12345678
```

Salida:

CurrentState	InstanceId	PreviousState
-----	-----	-----
Amazon.EC2.Model.InstanceState	i-12345678	Amazon.EC2.Model.InstanceState

- Para obtener más información sobre la API, consulte la referencia del [TerminateInstances](#) cmdlet AWS Tools for PowerShell .

## Python

### SDK para Python (Boto3)

#### Note

Hay más información al respecto. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

```
class InstanceWrapper:
    """Encapsulates Amazon Elastic Compute Cloud (Amazon EC2) instance
    actions."""

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

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

    def terminate(self):
```

```
"""
Terminates an instance and waits for it to be in a terminated state.
"""
if self.instance is None:
    logger.info("No instance to terminate.")
    return

instance_id = self.instance.id
try:
    self.instance.terminate()
    self.instance.wait_until_terminated()
    self.instance = None
except ClientError as err:
    logging.error(
        "Couldn't terminate instance %s. Here's why: %s: %s",
        instance_id,
        err.response["Error"]["Code"],
        err.response["Error"]["Message"],
    )
    raise
```

- Para obtener más información sobre la API, consulta [TerminateInstances](#) la AWS Referencia de API de SDK for Python (Boto3).

## Ruby

### SDK para Ruby

#### Note

Hay más información al respecto. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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."
  return true
rescue StandardError => e
  puts "Error terminating instance: #{e.message}"
  return false
end

# Example usage:
def run_me
  instance_id = ""
  region = ""
  # Print usage information and then stop.
  if ARGV[0] == "--help" || ARGV[0] == "-h"
    puts "Usage:  ruby ec2-ruby-example-terminate-instance-i-123abc.rb " \
      "INSTANCE_ID REGION "
    # Replace us-west-2 with the AWS Region you're using for Amazon EC2.
    puts "Example: ruby ec2-ruby-example-terminate-instance-i-123abc.rb " \
      "i-123abc us-west-2"
    exit 1
  # If no values are specified at the command prompt, use these default values.
  # Replace us-west-2 with the AWS Region you're using for Amazon EC2.
  elsif ARGV.count.zero?
```

```
instance_id = "i-123abc"
region = "us-west-2"
# Otherwise, use the values as specified at the command prompt.
else
  instance_id = ARGV[0]
  region = ARGV[1]
end

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

puts "Attempting to terminate instance '#{instance_id}' " \
      "(this might take a few minutes)..."
unless instance_terminated?(ec2_client, instance_id)
  puts "Could not terminate instance."
end
end

run_me if $PROGRAM_NAME == __FILE__
```

- Para obtener más información sobre la API, consulta [TerminateInstances](#) la Referencia AWS SDK for Ruby de la API.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte [Cree recursos de Amazon EC2 mediante un SDK AWS](#). En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo `UnassignPrivateIpAddresses` con un AWS SDK o CLI

En los siguientes ejemplos de código, se muestra cómo utilizar `UnassignPrivateIpAddresses`.

### CLI

#### AWS CLI

Para anular la asignación de una dirección IP privada secundaria de una interfaz de red

En este ejemplo, se anula la asignación de la dirección IP privada especificada de la interfaz de red especificada. Si el comando se ejecuta correctamente, no se muestra ningún resultado.

Comando:



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

- Para obtener más información sobre la API, consulte la Referencia [UnassignPrivateIpAddresses](#) de AWS CLI comandos.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: Este ejemplo anula la asignación de la dirección IP privada especificada de la interfaz de red especificada.

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

- Para obtener más información sobre la API, consulte la referencia de [UnassignPrivateIpAddresses AWS Tools for PowerShell](#) cmdlets.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Úselo **UnmonitorInstances** con un AWS SDK o CLI


En los siguientes ejemplos de código, se muestra cómo utilizar `UnmonitorInstances`.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en su contexto en el siguiente ejemplo de código:

- [Comience a utilizar instancias](#)

## C++

## SDK para C++

 Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

```
Aws::EC2::EC2Client ec2Client(clientConfiguration);
Aws::EC2::Model::UnmonitorInstancesRequest unrequest;
unrequest.AddInstanceIds(instanceId);
unrequest.SetDryRun(true);

auto undryRunOutcome = ec2Client.UnmonitorInstances(unrequest);
if (undryRunOutcome.IsSuccess()) {
    std::cerr
        << "Failed dry run to disable monitoring on instance. A dry run
should trigger an error."
        <<
        std::endl;
    return false;
}
else if (undryRunOutcome.GetError().GetErrorType() !=
    Aws::EC2::EC2Errors::DRY_RUN_OPERATION) {
    std::cout << "Failed dry run to disable monitoring on instance " <<
        instanceId << ": " << undryRunOutcome.GetError().GetMessage()
<<
        std::endl;
    return false;
}

unrequest.SetDryRun(false);
auto unmonitorInstancesOutcome = ec2Client.UnmonitorInstances(unrequest);
if (!unmonitorInstancesOutcome.IsSuccess()) {
    std::cout << "Failed to disable monitoring on instance " << instanceId
        << ": " << unmonitorInstancesOutcome.GetError().GetMessage() <<
        std::endl;
}
else {
    std::cout << "Successfully disable monitoring on instance " <<
```

```
        instanceId << std::endl;
    }
```

- Para obtener más información sobre la API, consulta [UnmonitorInstances](#) la Referencia AWS SDK for C++ de la API.

## CLI

### AWS CLI

Deshabilitar el monitoreo detallado de una instancia

Este comando de ejemplo deshabilita el monitoreo detallado de la instancia especificada.

Comando:

```
aws ec2 unmonitor-instances --instance-ids i-1234567890abcdef0
```

Salida:

```
{
  "InstanceMonitorings": [
    {
      "InstanceId": "i-1234567890abcdef0",
      "Monitoring": {
        "State": "disabling"
      }
    }
  ]
}
```

- Para obtener más información sobre la API, consulta [UnmonitorInstances](#) la Referencia de AWS CLI comandos.

## JavaScript

### SDK para JavaScript (v3)

#### Note

Hay más información. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

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

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

export const main = async () => {
  const command = new UnmonitorInstancesCommand({
    InstanceIds: ["i-09a3dfe7ae00e853f"],
  });

  try {
    const { InstanceMonitorings } = await client.send(command);
    const instanceMonitoringsList = InstanceMonitorings.map(
      (im) =>
        ` • Detailed monitoring state for ${im.InstanceId} is
        ${im.Monitoring.State}.`,
    );
    console.log("Monitoring status:");
    console.log(instanceMonitoringsList.join("\n"));
  } catch (err) {
    console.error(err);
  }
};
```

- Para obtener más información sobre la API, consulta [UnmonitorInstances](#) la Referencia AWS SDK for JavaScript de la API.

## PowerShell

### Herramientas para PowerShell

Ejemplo 1: en este ejemplo se deshabilita la supervisión detallada de la instancia especificada.

```
Stop-EC2InstanceMonitoring -InstanceId i-12345678
```

Salida:

```
InstanceId      Monitoring
-----
i-12345678     Amazon.EC2.Model.Monitoring
```

- Para obtener más información sobre la API, consulte la referencia [UnmonitorInstances](#) del AWS Tools for PowerShell cmdlet.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. [Cree recursos de Amazon EC2 mediante un SDK AWS](#) En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Escenarios para Amazon EC2 con SDK AWS

Los siguientes ejemplos de código muestran cómo implementar escenarios comunes en Amazon EC2 con AWS SDK. Estos escenarios muestran cómo llevar a cabo tareas específicas llamando a varias funciones dentro de Amazon EC2. Cada escenario incluye un enlace a GitHub, donde puede encontrar instrucciones sobre cómo configurar y ejecutar el código.

### Ejemplos

- [Cree y gestione un servicio resiliente mediante un AWS SDK](#)
- [Comience a utilizar las instancias de Amazon EC2 mediante un SDK AWS](#)

## Cree y gestione un servicio resiliente mediante un AWS SDK

Los siguientes ejemplos de código muestran cómo crear un servicio web con equilibrio de carga que muestre recomendaciones de libros, películas y canciones. El ejemplo muestra cómo responde el servicio a los errores y cómo reestructurarlo para aumentar la resiliencia cuando se produzcan errores.

- Utilice un grupo de Amazon EC2 Auto Scaling para crear instancias de Amazon Elastic Compute Cloud (Amazon EC2) basadas en una plantilla de lanzamiento y para mantener el número de instancias dentro de un rango específico.
- Administre y distribuya las solicitudes HTTP con Elastic Load Balancing.
- Supervise el estado de las instancias de un grupo de escalado automático y reenvíe las solicitudes solo a las instancias en buen estado.
- Ejecute un servidor web Python en cada instancia de EC2 para administrar las solicitudes HTTP. El servidor web responde con recomendaciones y comprobaciones de estado.
- Simule un servicio de recomendaciones con una tabla de Amazon DynamoDB.
- Controle la respuesta del servidor web a las solicitudes y las comprobaciones de estado actualizando AWS Systems Manager los parámetros.

## .NET

### AWS SDK for .NET

#### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

Ejecute el escenario interactivo en un símbolo del sistema.

```
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($"\\thttp://{endPoint}\\n");
    }
    else
    {
        Console.WriteLine(
            "\\nCouldn't get a successful response from the load balancer
endpoint. Troubleshoot by\\n"
            + "manually verifying that your VPC and security group are
configured correctly and that\\n"
            + "you can successfully make a GET request to the load balancer
endpoint:\\n");
        Console.WriteLine($"\\thttp://{endPoint}\\n");
    }
    Console.WriteLine(new string('-', 80));
    Console.WriteLine("Press Enter when you're ready to continue with the
demo.");
    if (interactive)
        Console.ReadLine();
    return true;
}

/// <summary>
/// Demonstrate the steps of the scenario.
/// </summary>
/// <param name="interactive">True to run as an interactive scenario.</param>
/// <returns>Async task.</returns>
public static async Task<bool> Demo(bool interactive)
{
    var ssmOnlyPolicy = Path.Join(_configuration["resourcePath"],
        "ssm_only_policy.json");

    Console.WriteLine(new string('-', 80));
    Console.WriteLine("Resetting parameters to starting values for demo.");
    await _smParameterWrapper.Reset();

    Console.WriteLine("\\nThis part of the demonstration shows how to toggle
different parts of the system\\n" +
        "to create situations where the web service fails, and
shows how using a resilient\\n" +
        "architecture can keep the web service running in spite
of these failures.");
    Console.WriteLine(new string('-', 88));
```

```
    Console.WriteLine("At the start, the load balancer endpoint returns
recommendations and reports that all targets are healthy.");
    if (interactive)
        await DemoActionChoices();

    Console.WriteLine($"The web service running on the EC2 instances gets
recommendations by querying a DynamoDB table.\n" +
        $"The table name is contained in a Systems Manager
parameter named '{_smParameterWrapper.TableParameter}'.\n" +
        $"To simulate a failure of the recommendation service,
let's set this parameter to name a non-existent table.\n");
    await
_smParameterWrapper.PutParameterByName(_smParameterWrapper.TableParameter,
    "this-is-not-a-table");
    Console.WriteLine("\nNow, sending a GET request to the load balancer
endpoint returns a failure code. But, the service reports as\n" +
        "healthy to the load balancer because shallow health
checks don't check for failure of the recommendation service.");
    if (interactive)
        await DemoActionChoices();

    Console.WriteLine("Instead of failing when the recommendation service
fails, the web service can return a static response.");
    Console.WriteLine("While this is not a perfect solution, it presents the
customer with a somewhat better experience than failure.");

    await
_smParameterWrapper.PutParameterByName(_smParameterWrapper.FailureResponseParameter,
    "static");

    Console.WriteLine("\nNow, sending a GET request to the load balancer
endpoint returns a static response.");
    Console.WriteLine("The service still reports as healthy because health
checks are still shallow.");
    if (interactive)
        await DemoActionChoices();

    Console.WriteLine("Let's reinstate the recommendation service.\n");
    await
_smParameterWrapper.PutParameterByName(_smParameterWrapper.TableParameter,
    _smParameterWrapper.TableName);
    Console.WriteLine(
        "\nLet's also substitute bad credentials for one of the instances in
the target group so that it can't\n" +
```

```
        "access the DynamoDB recommendation table.\n"
    );
    await _autoScalerWrapper.CreateInstanceProfileWithName(
        _autoScalerWrapper.BadCredsPolicyName,
        _autoScalerWrapper.BadCredsRoleName,
        _autoScalerWrapper.BadCredsProfileName,
        ssmOnlyPolicy,
        new List<string> { "AmazonSSMManagedInstanceCore" }
    );
    var instances = await
_autoScalerWrapper.GetInstancesByGroupName(_autoScalerWrapper.GroupName);
    var badInstanceId = instances.First();
    var instanceProfile = await
_autoScalerWrapper.GetInstanceProfile(badInstanceId);
    Console.WriteLine(
        $"Replacing the profile for instance {badInstanceId} with a profile
that contains\n" +
        "bad credentials...\n"
    );
    await _autoScalerWrapper.ReplaceInstanceProfile(
        badInstanceId,
        _autoScalerWrapper.BadCredsProfileName,
        instanceProfile.AssociationId
    );
    Console.WriteLine(
        "Now, sending a GET request to the load balancer endpoint returns
either a recommendation or a static response,\n" +
        "depending on which instance is selected by the load balancer.\n"
    );
    if (interactive)
        await DemoActionChoices();

    Console.WriteLine("\nLet's implement a deep health check. For this demo,
a deep health check tests whether");
    Console.WriteLine("the web service can access the DynamoDB table that it
depends on for recommendations. Note that");
    Console.WriteLine("the deep health check is only for ELB routing and not
for Auto Scaling instance health.");
    Console.WriteLine("This kind of deep health check is not recommended for
Auto Scaling instance health, because it");
    Console.WriteLine("risks accidental termination of all instances in the
Auto Scaling group when a dependent service fails.");
```

```
        Console.WriteLine("\nBy implementing deep health checks, the load
balancer can detect when one of the instances is failing");
        Console.WriteLine("and take that instance out of rotation.");

        await
_smParameterWrapper.PutParameterByName(_smParameterWrapper.HealthCheckParameter,
"deep");

        Console.WriteLine($" \nNow, checking target health indicates that the
instance with bad credentials ({badInstanceId})");
        Console.WriteLine("is unhealthy. Note that it might take a minute or two
for the load balancer to detect the unhealthy");
        Console.WriteLine("instance. Sending a GET request to the load balancer
endpoint always returns a recommendation, because");
        Console.WriteLine("the load balancer takes unhealthy instances out of its
rotation.");

        if (interactive)
            await DemoActionChoices();

        Console.WriteLine("\nBecause the instances in this demo are controlled by
an auto scaler, the simplest way to fix an unhealthy");
        Console.WriteLine("instance is to terminate it and let the auto scaler
start a new instance to replace it.");

        await _autoScalerWrapper.TryTerminateInstanceById(badInstanceId);

        Console.WriteLine($" \nEven while the instance is terminating and the new
instance is starting, sending a GET");
        Console.WriteLine("request to the web service continues to get a
successful recommendation response because");
        Console.WriteLine("starts and reports as healthy, it is included in the
load balancing rotation.");
        Console.WriteLine("Note that terminating and replacing an instance
typically takes several minutes, during which time you");
        Console.WriteLine("can see the changing health check status until the new
instance is running and healthy.");

        if (interactive)
            await DemoActionChoices();

        Console.WriteLine("\nIf the recommendation service fails now, deep health
checks mean all instances report as unhealthy.");
```



```

        await
        _smParameterWrapper.PutParameterByName(_smParameterWrapper.TableParameter,
        "this-is-not-a-table");

        Console.WriteLine($"\\nWhen all instances are unhealthy, the load balancer
        continues to route requests even to");
        Console.WriteLine("unhealthy instances, allowing them to fail open and
        return a static response rather than fail");
        Console.WriteLine("closed and report failure to the customer.");

        if (interactive)
            await DemoActionChoices();
        await _smParameterWrapper.Reset();

        Console.WriteLine(new string('-', 80));
        return true;
    }

    /// <summary>
    /// Clean up the resources from the scenario.
    /// </summary>
    /// <param name="interactive">True to ask the user for cleanup.</param>
    /// <returns>Async task.</returns>
    public static async Task<bool> DestroyResources(bool interactive)
    {
        Console.WriteLine(new string('-', 80));
        Console.WriteLine(
            "To keep things tidy and to avoid unwanted charges on your account,
we can clean up all AWS resources\\n" +
            "that were created for this demo."
        );

        if (!interactive || GetYesNoResponse("Do you want to clean up all demo
resources? (y/n) "))
        {
            await
            _elasticLoadBalancerWrapper.DeleteLoadBalancerByName(_elasticLoadBalancerWrapper.LoadBal
            await
            _elasticLoadBalancerWrapper.DeleteTargetGroupByName(_elasticLoadBalancerWrapper.TargetGr
            await
            _autoScalerWrapper.TerminateAndDeleteAutoScalingGroupWithName(_autoScalerWrapper.GroupNa
            await
            _autoScalerWrapper.DeleteKeyPairByName(_autoScalerWrapper.KeyPairName);

```

```

        await
        _autoScalerWrapper.DeleteTemplateByName(_autoScalerWrapper.LaunchTemplateName);
        await _autoScalerWrapper.DeleteInstanceProfile(
            _autoScalerWrapper.BadCredsProfileName,
            _autoScalerWrapper.BadCredsRoleName
        );
        await
        _recommendations.DestroyDatabaseByName(_recommendations.TableName);
    }
    else
    {
        Console.WriteLine(
            "Ok, we'll leave the resources intact.\n" +
            "Don't forget to delete them when you're done with them or you
            might incur unexpected charges."
        );
    }

    Console.WriteLine(new string('-', 80));
    return true;
}

```

Cree una clase que agrupe las acciones de escalado automático y Amazon EC2.

```

/// <summary>
/// Encapsulates Amazon EC2 Auto Scaling and EC2 management methods.
/// </summary>
public class AutoScalerWrapper
{
    private readonly IAmazonAutoScaling _amazonAutoScaling;
    private readonly IAmazonEC2 _amazonEc2;
    private readonly IAmazonSimpleSystemsManagement _amazonSsm;
    private readonly IAmazonIdentityManagementService _amazonIam;

    private readonly string _instanceType = "";
    private readonly string _amiParam = "";
    private readonly string _launchTemplateName = "";
    private readonly string _groupName = "";
    private readonly string _instancePolicyName = "";
    private readonly string _instanceRoleName = "";
    private readonly string _instanceProfileName = "";
    private readonly string _badCredsProfileName = "";
}

```

```
private readonly string _badCredsRoleName = "";
private readonly string _badCredsPolicyName = "";
private readonly string _keyPairName = "";

public string GroupName => _groupName;
public string KeyPairName => _keyPairName;
public string LaunchTemplateName => _launchTemplateName;
public string InstancePolicyName => _instancePolicyName;
public string BadCredsProfileName => _badCredsProfileName;
public string BadCredsRoleName => _badCredsRoleName;
public string BadCredsPolicyName => _badCredsPolicyName;

/// <summary>
/// Constructor for the AutoScalerWrapper.
/// </summary>
/// <param name="amazonAutoScaling">The injected AutoScaling client.</param>
/// <param name="amazonEc2">The injected EC2 client.</param>
/// <param name="amazonIam">The injected IAM client.</param>
/// <param name="amazonSsm">The injected SSM client.</param>
public AutoScalerWrapper(
    IAmazonAutoScaling amazonAutoScaling,
    IAmazonEC2 amazonEc2,
    IAmazonSimpleSystemsManagement amazonSsm,
    IAmazonIdentityManagementService amazonIam,
    IConfiguration configuration)
{
    _amazonAutoScaling = amazonAutoScaling;
    _amazonEc2 = amazonEc2;
    _amazonSsm = amazonSsm;
    _amazonIam = amazonIam;

    var prefix = configuration["resourcePrefix"];
    _instanceType = configuration["instanceType"];
    _amiParam = configuration["amiParam"];

    _launchTemplateName = prefix + "-template";
    _groupName = prefix + "-group";
    _instancePolicyName = prefix + "-pol";
    _instanceRoleName = prefix + "-role";
    _instanceProfileName = prefix + "-prof";
    _badCredsPolicyName = prefix + "-bc-pol";
    _badCredsRoleName = prefix + "-bc-role";
    _badCredsProfileName = prefix + "-bc-prof";
    _keyPairName = prefix + "-key-pair";
}
```

```

}

/// <summary>
/// Create a policy, role, and profile that is associated with instances with
a specified name.
/// An instance's associated profile defines a role that is assumed by the
/// instance. The role has attached policies that specify the AWS permissions
granted to
/// clients that run on the instance.
/// </summary>
/// <param name="policyName">Name to use for the policy.</param>
/// <param name="roleName">Name to use for the role.</param>
/// <param name="profileName">Name to use for the profile.</param>
/// <param name="ssmOnlyPolicyFile">Path to a policy file for SSM.</param>
/// <param name="awsManagedPolicies">AWS Managed policies to be attached to
the role.</param>
/// <returns>The Arn of the profile.</returns>
public async Task<string> CreateInstanceProfileWithName(
    string policyName,
    string roleName,
    string profileName,
    string ssmOnlyPolicyFile,
    List<string>? awsManagedPolicies = null)
{
    var assumeRoleDoc = "{" +
        "\"Version\": \"2012-10-17\", " +
        "\"Statement\": [{" +
            "\"Effect\": \"Allow\", " +
            "\"Principal\": {" +
            "\"Service\": [" +
                "\"ec2.amazonaws.com\"" +
            "]" +
            "}, " +
            "\"Action\": \"sts:AssumeRole\"" +
        "}] " +
    "};

    var policyDocument = await File.ReadAllTextAsync(ssmOnlyPolicyFile);

    var policyArn = "";

    try
    {

```

```
var createPolicyResult = await _amazonIam.CreatePolicyAsync(
    new CreatePolicyRequest
    {
        PolicyName = policyName,
        PolicyDocument = policyDocument
    });
policyArn = createPolicyResult.Policy.Arn;
}
catch (EntityAlreadyExistsException)
{
    // The policy already exists, so we look it up to get the Arn.
    var policiesPaginator = _amazonIam.Paginators.ListPolicies(
        new ListPoliciesRequest()
        {
            Scope = PolicyScopeType.Local
        });
    // Get the entire list using the paginator.
    await foreach (var policy in policiesPaginator.Policies)
    {
        if (policy.PolicyName.Equals(policyName))
        {
            policyArn = policy.Arn;
        }
    }

    if (policyArn == null)
    {
        throw new InvalidOperationException("Policy not found");
    }
}

try
{
    await _amazonIam.CreateRoleAsync(new CreateRoleRequest()
    {
        RoleName = roleName,
        AssumeRolePolicyDocument = assumeRoleDoc,
    });
    await _amazonIam.AttachRolePolicyAsync(new AttachRolePolicyRequest()
    {
        RoleName = roleName,
        PolicyArn = policyArn
    });
    if (awsManagedPolicies != null)
```

```
        {
            foreach (var awsPolicy in awsManagedPolicies)
            {
                await _amazonIam.AttachRolePolicyAsync(new
AttachRolePolicyRequest()
                {
                    PolicyArn = $"arn:aws:iam::aws:policy/{awsPolicy}",
                    RoleName = roleName
                });
            }
        }
    }
    catch (EntityAlreadyExistsException)
    {
        Console.WriteLine("Role already exists.");
    }

    string profileArn = "";
    try
    {
        var profileCreateResponse = await
_amazonIam.CreateInstanceProfileAsync(
            new CreateInstanceProfileRequest()
            {
                InstanceProfileName = profileName
            });
        // Allow time for the profile to be ready.
        profileArn = profileCreateResponse.InstanceProfile.Arn;
        Thread.Sleep(10000);
        await _amazonIam.AddRoleToInstanceProfileAsync(
            new AddRoleToInstanceProfileRequest()
            {
                InstanceProfileName = profileName,
                RoleName = roleName
            });
    }
    catch (EntityAlreadyExistsException)
    {
        Console.WriteLine("Policy already exists.");
        var profileGetResponse = await _amazonIam.GetInstanceProfileAsync(
            new GetInstanceProfileRequest()
            {
                InstanceProfileName = profileName
            });
    }
}
```

```
        });
        profileArn = profileGetResponse.InstanceProfile.Arn;
    }
    return profileArn;
}

/// <summary>
/// Create a new key pair and save the file.
/// </summary>
/// <param name="newKeyName">The name of the new key pair.</param>
/// <returns>Async task.</returns>
public async Task CreateKeyPair(string newKeyName)
{
    try
    {
        var keyResponse = await _amazonEc2.CreateKeyPairAsync(
            new CreateKeyPairRequest() { KeyName = newKeyName });
        await File.WriteAllTextAsync($"{newKeyName}.pem",
            keyResponse.KeyPair.KeyMaterial);
        Console.WriteLine($"Created key pair {newKeyName}.");
    }
    catch (AlreadyExistsException)
    {
        Console.WriteLine("Key pair already exists.");
    }
}

/// <summary>
/// Delete the key pair and file by name.
/// </summary>
/// <param name="deleteKeyName">The key pair to delete.</param>
/// <returns>Async task.</returns>
public async Task DeleteKeyPairByName(string deleteKeyName)
{
    try
    {
        await _amazonEc2.DeleteKeyPairAsync(
            new DeleteKeyPairRequest() { KeyName = deleteKeyName });
        File.Delete($"{deleteKeyName}.pem");
    }
    catch (FileNotFoundException)
    {
        Console.WriteLine($"Key pair {deleteKeyName} not found.");
    }
}
```

```
}

/// <summary>
/// Creates an Amazon EC2 launch template to use with Amazon EC2 Auto
Scaling.
/// The launch template specifies a Bash script in its user data field that
runs after
/// the instance is started. This script installs the Python packages and
starts a Python
/// web server on the instance.
/// </summary>
/// <param name="startupScriptPath">The path to a Bash script file that is
run.</param>
/// <param name="instancePolicyPath">The path to a permissions policy to
create and attach to the profile.</param>
/// <returns>The template object.</returns>
public async Task<Amazon.EC2.Model.LaunchTemplate> CreateTemplate(string
startupScriptPath, string instancePolicyPath)
{
    await CreateKeyPair(_keyPairName);
    await CreateInstanceProfileWithName(_instancePolicyName,
_instanceRoleName, _instanceProfileName, instancePolicyPath);

    var startServerText = await File.ReadAllTextAsync(startupScriptPath);
    var plainTextBytes = System.Text.Encoding.UTF8.GetBytes(startServerText);

    var amiLatest = await _amazonSsm.GetParameterAsync(
        new GetParameterRequest() { Name = _amiParam });
    var amiId = amiLatest.Parameter.Value;
    var launchTemplateResponse = await _amazonEc2.CreateLaunchTemplateAsync(
        new CreateLaunchTemplateRequest()
        {
            LaunchTemplateName = _launchTemplateName,
            LaunchTemplateData = new RequestLaunchTemplateData()
            {
                InstanceType = _instanceType,
                ImageId = amiId,
                IamInstanceProfile =
                    new
LaunchTemplateIamInstanceProfileSpecificationRequest()
                    {
                        Name = _instanceProfileName
                    },
            },
        },
    );
}
```



```

        KeyName = _keyPairName,
        UserData = System.Convert.ToBase64String(plainTextBytes)
    }
});
return launchTemplateResponse.LaunchTemplate;

}

/// <summary>
/// Get a list of Availability Zones in the AWS Region of the Amazon EC2
Client.
/// </summary>
/// <returns>A list of availability zones.</returns>
public async Task<List<string>> DescribeAvailabilityZones()
{
    var zoneResponse = await _amazonEc2.DescribeAvailabilityZonesAsync(
        new DescribeAvailabilityZonesRequest());
    return zoneResponse.AvailabilityZones.Select(z => z.ZoneName).ToList();
}

/// <summary>
/// Create an EC2 Auto Scaling group of a specified size and name.
/// </summary>
/// <param name="groupSize">The size for the group.</param>
/// <param name="groupName">The name for the group.</param>
/// <param name="availabilityZones">The availability zones for the group.</
param>
/// <returns>Async task.</returns>
public async Task CreateGroupOfSize(int groupSize, string groupName,
List<string> availabilityZones)
{
    try
    {
        await _amazonAutoScaling.CreateAutoScalingGroupAsync(
            new CreateAutoScalingGroupRequest()
            {
                AutoScalingGroupName = groupName,
                AvailabilityZones = availabilityZones,
                LaunchTemplate =
                    new
Amazon.AutoScaling.Model.LaunchTemplateSpecification()
                    {
                        LaunchTemplateName = _launchTemplateName,
                        Version = "$Default"
                    }
            }
        );
    }
}

```

```
        },
        MaxSize = groupSize,
        MinSize = groupSize
    });
    Console.WriteLine($"Created EC2 Auto Scaling group {groupName} with
size {groupSize}.");
}
catch (EntityAlreadyExistsException)
{
    Console.WriteLine($"EC2 Auto Scaling group {groupName} already
exists.");
}
}

/// <summary>
/// Get the default VPC for the account.
/// </summary>
/// <returns>The default VPC object.</returns>
public async Task<Vpc> GetDefaultVpc()
{
    var vpcResponse = await _amazonEc2.DescribeVpcsAsync(
        new DescribeVpcsRequest()
        {
            Filters = new List<Amazon.EC2.Model.Filter>()
            {
                new ("is-default", new List<string>() { "true" })
            }
        });
    return vpcResponse.Vpcs[0];
}

/// <summary>
/// Get all the subnets for a Vpc in a set of availability zones.
/// </summary>
/// <param name="vpcId">The Id of the Vpc.</param>
/// <param name="availabilityZones">The list of availability zones.</param>
/// <returns>The collection of subnet objects.</returns>
public async Task<List<Subnet>> GetAllVpcSubnetsForZones(string vpcId,
List<string> availabilityZones)
{
    var subnets = new List<Subnet>();
    var subnetPaginator = _amazonEc2.Paginators.DescribeSubnets(
        new DescribeSubnetsRequest()
        {
```

```
        Filters = new List<Amazon.EC2.Model.Filter>()
        {
            new ("vpc-id", new List<string>() { vpcId}),
            new ("availability-zone", availabilityZones),
            new ("default-for-az", new List<string>() { "true" })
        }
    });

    // Get the entire list using the paginator.
    await foreach (var subnet in subnetPaginator.Subnets)
    {
        subnets.Add(subnet);
    }

    return subnets;
}

/// <summary>
/// Delete a launch template by name.
/// </summary>
/// <param name="templateName">The name of the template to delete.</param>
/// <returns>Async task.</returns>
public async Task DeleteTemplateByName(string templateName)
{
    try
    {
        await _amazonEc2.DeleteLaunchTemplateAsync(
            new DeleteLaunchTemplateRequest()
            {
                LaunchTemplateName = templateName
            });
    }
    catch (AmazonClientException)
    {
        Console.WriteLine($"Unable to delete template {templateName}.");
    }
}

/// <summary>
/// Detaches a role from an instance profile, detaches policies from the
role,
/// and deletes all the resources.
/// </summary>
/// <param name="profileName">The name of the profile to delete.</param>
```

```
/// <param name="roleName">The name of the role to delete.</param>
/// <returns>Async task.</returns>
public async Task DeleteInstanceProfile(string profileName, string roleName)
{
    try
    {
        await _amazonIam.RemoveRoleFromInstanceProfileAsync(
            new RemoveRoleFromInstanceProfileRequest()
            {
                InstanceProfileName = profileName,
                RoleName = roleName
            });
        await _amazonIam.DeleteInstanceProfileAsync(
            new DeleteInstanceProfileRequest() { InstanceProfileName =
profileName });
        var attachedPolicies = await
_amazonIam.ListAttachedRolePoliciesAsync(
            new ListAttachedRolePoliciesRequest() { RoleName = roleName });
        foreach (var policy in attachedPolicies.AttachedPolicies)
        {
            await _amazonIam.DetachRolePolicyAsync(
                new DetachRolePolicyRequest()
                {
                    RoleName = roleName,
                    PolicyArn = policy.PolicyArn
                });
            // Delete the custom policies only.
            if (!policy.PolicyArn.StartsWith("arn:aws:iam::aws"))
            {
                await _amazonIam.DeletePolicyAsync(
                    new Amazon.IdentityManagement.Model.DeletePolicyRequest()
                    {
                        PolicyArn = policy.PolicyArn
                    });
            }
        }

        await _amazonIam.DeleteRoleAsync(
            new DeleteRoleRequest() { RoleName = roleName });
    }
    catch (NoSuchEntityException)
    {
        Console.WriteLine($"Instance profile {profileName} does not exist.");
    }
}
```

```
    }

    /// <summary>
    /// Gets data about the instances in an EC2 Auto Scaling group by its group
    name.
    /// </summary>
    /// <param name="group">The name of the auto scaling group.</param>
    /// <returns>A collection of instance Ids.</returns>
    public async Task<IEnumerable<string>> GetInstancesByGroupName(string group)
    {
        var instanceResponse = await
        _amazonAutoScaling.DescribeAutoScalingGroupsAsync(
            new DescribeAutoScalingGroupsRequest()
            {
                AutoScalingGroupNames = new List<string>() { group }
            });
        var instanceIds = instanceResponse.AutoScalingGroups.SelectMany(
            g => g.Instances.Select(i => i.InstanceId));
        return instanceIds;
    }

    /// <summary>
    /// Get the instance profile association data for an instance.
    /// </summary>
    /// <param name="instanceId">The Id of the instance.</param>
    /// <returns>Instance profile associations data.</returns>
    public async Task<IamInstanceProfileAssociation> GetInstanceProfile(string
    instanceId)
    {
        var response = await
        _amazonEc2.DescribeIamInstanceProfileAssociationsAsync(
            new DescribeIamInstanceProfileAssociationsRequest()
            {
                Filters = new List<Amazon.EC2.Model.Filter>()
                {
                    new ("instance-id", new List<string>() { instanceId })
                },
            });
        return response.IamInstanceProfileAssociations[0];
    }

    /// <summary>
    /// Replace the profile associated with a running instance. After the profile
    is replaced, the instance
```

```
    /// is rebooted to ensure that it uses the new profile. When the instance is
    /// ready, Systems Manager is
    /// used to restart the Python web server.
    /// </summary>
    /// <param name="instanceId">The Id of the instance to update.</param>
    /// <param name="credsProfileName">The name of the new profile to associate
    /// with the specified instance.</param>
    /// <param name="associationId">The Id of the existing profile association
    /// for the instance.</param>
    /// <returns>Async task.</returns>
    public async Task ReplaceInstanceProfile(string instanceId, string
    credsProfileName, string associationId)
    {
        await _amazonEc2.ReplaceIamInstanceProfileAssociationAsync(
            new ReplaceIamInstanceProfileAssociationRequest()
            {
                AssociationId = associationId,
                IamInstanceProfile = new IamInstanceProfileSpecification()
                {
                    Name = credsProfileName
                }
            }
        );
        // Allow time before resetting.
        Thread.Sleep(25000);
        var instanceReady = false;
        var retries = 5;
        while (retries-- > 0 && !instanceReady)
        {
            await _amazonEc2.RebootInstancesAsync(
                new RebootInstancesRequest(new List<string>() { instanceId }));
            Thread.Sleep(10000);

            var instancesPaginator =
            _amazonSsm.Paginators.DescribeInstanceInformation(
                new DescribeInstanceInformationRequest());
            // Get the entire list using the paginator.
            await foreach (var instance in
            instancesPaginator.InstanceInformationList)
            {
                instanceReady = instance.InstanceId == instanceId;
                if (instanceReady)
                {
                    break;
                }
            }
        }
    }
}
```

```

    }
}
Console.WriteLine($"Sending restart command to instance {instanceId}");
await _amazonSsm.SendCommandAsync(
    new SendCommandRequest()
    {
        InstanceIds = new List<string>() { instanceId },
        DocumentName = "AWS-RunShellScript",
        Parameters = new Dictionary<string, List<string>>()
        {
            {"commands", new List<string>() { "cd / && sudo python3
server.py 80" }}
        }
    });
Console.WriteLine($"Restarted the web server on instance {instanceId}");
}

/// <summary>
/// Try to terminate an instance by its Id.
/// </summary>
/// <param name="instanceId">The Id of the instance to terminate.</param>
/// <returns>Async task.</returns>
public async Task TryTerminateInstanceById(string instanceId)
{
    var stopping = false;
    Console.WriteLine($"Stopping {instanceId}...");
    while (!stopping)
    {
        try
        {
            await
            _amazonAutoScaling.TerminateInstanceInAutoScalingGroupAsync(
                new TerminateInstanceInAutoScalingGroupRequest()
                {
                    InstanceId = instanceId,
                    ShouldDecrementDesiredCapacity = false
                });
            stopping = true;
        }
        catch (ScalingActivityInProgressException)
        {
            Console.WriteLine($"Scaling activity in progress for
{instanceId}. Waiting...");
            Thread.Sleep(10000);
        }
    }
}

```

```
    }
  }
}

/// <summary>
/// Tries to delete the EC2 Auto Scaling group. If the group is in use or in
progress,
/// waits and retries until the group is successfully deleted.
/// </summary>
/// <param name="groupName">The name of the group to try to delete.</param>
/// <returns>Async task.</returns>
public async Task TryDeleteGroupByName(string groupName)
{
    var stopped = false;
    while (!stopped)
    {
        try
        {
            await _amazonAutoScaling.DeleteAutoScalingGroupAsync(
                new DeleteAutoScalingGroupRequest()
                {
                    AutoScalingGroupName = groupName
                });
            stopped = true;
        }
        catch (Exception e)
            when ((e is ScalingActivityInProgressException)
                || (e is Amazon.AutoScaling.Model.ResourceInUseException))
        {
            Console.WriteLine($"Some instances are still running.
Waiting...");
            Thread.Sleep(10000);
        }
    }
}

/// <summary>
/// Terminate instances and delete the Auto Scaling group by name.
/// </summary>
/// <param name="groupName">The name of the group to delete.</param>
/// <returns>Async task.</returns>
public async Task TerminateAndDeleteAutoScalingGroupWithName(string
groupName)
{
```



```

    var describeGroupsResponse = await
    _amazonAutoScaling.DescribeAutoScalingGroupsAsync(
        new DescribeAutoScalingGroupsRequest()
        {
            AutoScalingGroupNames = new List<string>() { groupName }
        });
    if (describeGroupsResponse.AutoScalingGroups.Any())
    {
        // Update the size to 0.
        await _amazonAutoScaling.UpdateAutoScalingGroupAsync(
            new UpdateAutoScalingGroupRequest()
            {
                AutoScalingGroupName = groupName,
                MinSize = 0
            });
        var group = describeGroupsResponse.AutoScalingGroups[0];
        foreach (var instance in group.Instances)
        {
            await TryTerminateInstanceById(instance.InstanceId);
        }

        await TryDeleteGroupByName(groupName);
    }
    else
    {
        Console.WriteLine($"No groups found with name {groupName}.");
    }
}

/// <summary>
/// Get the default security group for a specified Vpc.
/// </summary>
/// <param name="vpc">The Vpc to search.</param>
/// <returns>The default security group.</returns>
public async Task<SecurityGroup> GetDefaultSecurityGroupForVpc(Vpc vpc)
{
    var groupResponse = await _amazonEc2.DescribeSecurityGroupsAsync(
        new DescribeSecurityGroupsRequest()
        {
            Filters = new List<Amazon.EC2.Model.Filter>()
            {
                new ("group-name", new List<string>() { "default" }),
                new ("vpc-id", new List<string>() { vpc.VpcId })
            }
        }
    );
}

```

```
        }
    });
    return groupResponse.SecurityGroups[0];
}

/// <summary>
/// Verify the default security group of a Vpc allows ingress from the
calling computer.
/// This can be done by allowing ingress from this computer's IP address.
/// In some situations, such as connecting from a corporate network, you must
instead specify
/// a prefix list Id. You can also temporarily open the port to any IP
address while running this example.
/// If you do, be sure to remove public access when you're done.
/// </summary>
/// <param name="vpc">The group to check.</param>
/// <param name="port">The port to verify.</param>
/// <param name="ipAddress">This computer's IP address.</param>
/// <returns>True if the ip address is allowed on the group.</returns>
public bool VerifyInboundPortForGroup(SecurityGroup group, int port, string
ipAddress)
{
    var portIsOpen = false;
    foreach (var ipPermission in group.IpPermissions)
    {
        if (ipPermission.FromPort == port)
        {
            foreach (var ipRange in ipPermission.Ipv4Ranges)
            {
                var cidr = ipRange.CidrIp;
                if (cidr.StartsWith(ipAddress) || cidr == "0.0.0.0/0")
                {
                    portIsOpen = true;
                }
            }

            if (ipPermission.PrefixListIds.Any())
            {
                portIsOpen = true;
            }

            if (!portIsOpen)
            {
```

```

        Console.WriteLine("The inbound rule does not appear to be
open to either this computer's IP\n" +
                        "address, to all IP addresses (0.0.0.0/0),
or to a prefix list ID.");
    }
    else
    {
        break;
    }
}
}

return portIsOpen;
}

/// <summary>
/// Add an ingress rule to the specified security group that allows access on
the
/// specified port from the specified IP address.
/// </summary>
/// <param name="groupId">The Id of the security group to modify.</param>
/// <param name="port">The port to open.</param>
/// <param name="ipAddress">The IP address to allow access.</param>
/// <returns>Async task.</returns>
public async Task OpenInboundPort(string groupId, int port, string ipAddress)
{
    await _amazonEc2.AuthorizeSecurityGroupIngressAsync(
        new AuthorizeSecurityGroupIngressRequest()
        {
            GroupId = groupId,
            IpPermissions = new List<IpPermission>()
            {
                new IpPermission()
                {
                    FromPort = port,
                    ToPort = port,
                    IpProtocol = "tcp",
                    Ipv4Ranges = new List<IpRange>()
                    {
                        new IpRange() { CidrIp = $"{ipAddress}/32" }
                    }
                }
            }
        });
}

```

```

    }

    /// <summary>
    /// Attaches an Elastic Load Balancing (ELB) target group to this EC2 Auto
    Scaling group.
    /// The
    /// </summary>
    /// <param name="autoScalingGroupName">The name of the Auto Scaling group.</
param>
    /// <param name="targetGroupArn">The Arn for the target group.</param>
    /// <returns>Async task.</returns>
    public async Task AttachLoadBalancerToGroup(string autoScalingGroupName,
string targetGroupArn)
    {
        await _amazonAutoScaling.AttachLoadBalancerTargetGroupsAsync(
            new AttachLoadBalancerTargetGroupsRequest()
            {
                AutoScalingGroupName = autoScalingGroupName,
                TargetGroupARNs = new List<string>() { targetGroupArn }
            });
    }
}

```

Cree una clase que resuma las acciones de 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);
        }
    }
}
}

```

Cree una clase que utilice DynamoDB para simular un servicio de recomendaciones.

```

/// <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");
    }
}
}

```

Cree una clase que agrupe las acciones de 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 obtener información sobre la API, consulte los siguientes temas en la referencia de la API de AWS SDK for .NET .
  - [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](#)

## Java

### SDK para Java 2.x

#### Note

Hay más información GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

Ejecute el escenario interactivo en un símbolo del sistema.

```
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 wasSuccessful =
        loadBalancer.verifyLoadBalancerEndpoint(elbDnsName);
        if (!wasSuccessful) {
            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);
}
}

```

Cree una clase que agrupe las acciones de escalado automático y Amazon EC2.

```

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.");
}
}

```

Cree una clase que resuma las acciones de 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 "";
}
}
```

Cree una clase que utilice DynamoDB para simular un servicio de recomendaciones.

```
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);
}
}
}

```

Cree una clase que agrupe las acciones de 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 obtener información sobre la API, consulte los siguientes temas en la referencia de la API de AWS SDK for Java 2.x .
  - [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](#)

## JavaScript

### SDK para JavaScript (v3)

#### Note

Hay más información. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

Ejecute el escenario interactivo en un símbolo del sistema.

```
#!/usr/bin/env node
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: Apache-2.0

import {
```

```
Scenario,
parseScenarioArgs,
} from "@aws-doc-sdk-examples/lib/scenario/index.js";

/**
 * The workflow steps are split into three stages:
 * - deploy
 * - demo
 * - destroy
 *
 * Each of these stages has a corresponding file prefixed with steps-*.
 */
import { deploySteps } from "./steps-deploy.js";
import { demoSteps } from "./steps-demo.js";
import { destroySteps } from "./steps-destroy.js";

/**
 * The context is passed to every scenario. Scenario steps
 * will modify the context.
 */
const context = {};

/**
 * Three Scenarios are created for the workflow. A Scenario is an orchestration
class
 * that simplifies running a series of steps.
 */
export const scenarios = {
  // Deploys all resources necessary for the workflow.
  deploy: new Scenario("Resilient Workflow - Deploy", deploySteps, context),
  // Demonstrates how a fragile web service can be made more resilient.
  demo: new Scenario("Resilient Workflow - Demo", demoSteps, context),
  // Destroys the resources created for the workflow.
  destroy: new Scenario("Resilient Workflow - Destroy", destroySteps, context),
};

// Call function if run directly
import { fileURLToPath } from "url";

if (process.argv[1] === fileURLToPath(import.meta.url)) {
  parseScenarioArgs(scenarios);
}
```

## Cree los pasos para implementar todos los recursos.

```
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: Apache-2.0
import { join } from "node:path";
import { readFileSync, writeFileSync } from "node:fs";
import axios from "axios";

import {
  BatchWriteItemCommand,
  CreateTableCommand,
  DynamoDBClient,
  waitUntilTableExists,
} from "@aws-sdk/client-dynamodb";
import {
  EC2Client,
  CreateKeyPairCommand,
  CreateLaunchTemplateCommand,
  DescribeAvailabilityZonesCommand,
  DescribeVpcsCommand,
  DescribeSubnetsCommand,
  DescribeSecurityGroupsCommand,
  AuthorizeSecurityGroupIngressCommand,
} from "@aws-sdk/client-ec2";
import {
  IAMClient,
  CreatePolicyCommand,
  CreateRoleCommand,
  CreateInstanceProfileCommand,
  AddRoleToInstanceProfileCommand,
  AttachRolePolicyCommand,
  waitUntilInstanceProfileExists,
} from "@aws-sdk/client-iam";
import { SSMClient, GetParameterCommand } from "@aws-sdk/client-ssm";
import {
  CreateAutoScalingGroupCommand,
  AutoScalingClient,
  AttachLoadBalancerTargetGroupsCommand,
} from "@aws-sdk/client-auto-scaling";
import {
  CreateListenerCommand,
  CreateLoadBalancerCommand,
  CreateTargetGroupCommand,
  ElasticLoadBalancingV2Client,
}
```

```
    waitUntilLoadBalancerAvailable,
  } from "@aws-sdk/client-elastic-load-balancing-v2";

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

import { MESSAGES, NAMES, RESOURCES_PATH, ROOT } from "./constants.js";
import { initParamsSteps } from "./steps-reset-params.js";

/**
 * @type {import('@aws-doc-sdk-examples/lib/scenario.js').Step[]}
 */
export const deploySteps = [
  new ScenarioOutput("introduction", MESSAGES.introduction, { header: true }),
  new ScenarioInput("confirmDeployment", MESSAGES.confirmDeployment, {
    type: "confirm",
  }),
  new ScenarioAction(
    "handleConfirmDeployment",
    (c) => c.confirmDeployment === false && process.exit(),
  ),
  new ScenarioOutput(
    "creatingTable",
    MESSAGES.creatingTable.replace("${TABLE_NAME}", NAMES.tableName),
  ),
  new ScenarioAction("createTable", async () => {
    const client = new DynamoDBClient({});
    await client.send(
      new CreateTableCommand({
        TableName: NAMES.tableName,
        ProvisionedThroughput: {
          ReadCapacityUnits: 5,
          WriteCapacityUnits: 5,
        },
        AttributeDefinitions: [
          {
            AttributeName: "MediaType",
            AttributeType: "S",
          },
          {

```



```

        AttributeName: "ItemId",
        AttributeType: "N",
    },
],
KeySchema: [
    {
        AttributeName: "MediaTypeId",
        KeyType: "HASH",
    },
    {
        AttributeName: "ItemId",
        KeyType: "RANGE",
    },
],
}),
);
await waitUntilTableExists({ client }, { TableName: NAMES.tableName });
}),
new ScenarioOutput(
    "createdTable",
    MESSAGES.createdTable.replace("${TABLE_NAME}", NAMES.tableName),
),
new ScenarioOutput(
    "populatingTable",
    MESSAGES.populatingTable.replace("${TABLE_NAME}", NAMES.tableName),
),
new ScenarioAction("populateTable", () => {
    const client = new DynamoDBClient({});
    /**
     * @type {{ default: import("@aws-sdk/client-dynamodb").PutRequest['Item']
[] }}
    */
    const recommendations = JSON.parse(
        readFileSync(join(RESOURCES_PATH, "recommendations.json")),
    );

    return client.send(
        new BatchWriteItemCommand({
            RequestItems: {
                [NAMES.tableName]: recommendations.map((item) => ({
                    PutRequest: { Item: item },
                })),
            },
        }),
    ),
});

```

```
);
}),
new ScenarioOutput(
  "populatedTable",
  MESSAGES.populatedTable.replace("${TABLE_NAME}", NAMES.tableName),
),
new ScenarioOutput(
  "creatingKeyPair",
  MESSAGES.creatingKeyPair.replace("${KEY_PAIR_NAME}", NAMES.keyPairName),
),
new ScenarioAction("createKeyPair", async () => {
  const client = new EC2Client({});
  const { KeyMaterial } = await client.send(
    new CreateKeyPairCommand({
      KeyName: NAMES.keyPairName,
    }),
  );
  writeFileSync(`${NAMES.keyPairName}.pem`, KeyMaterial, { mode: 0o600 });
}),
new ScenarioOutput(
  "createdKeyPair",
  MESSAGES.createdKeyPair.replace("${KEY_PAIR_NAME}", NAMES.keyPairName),
),
new ScenarioOutput(
  "creatingInstancePolicy",
  MESSAGES.creatingInstancePolicy.replace(
    "${INSTANCE_POLICY_NAME}",
    NAMES.instancePolicyName,
  ),
),
new ScenarioAction("createInstancePolicy", async (state) => {
  const client = new IAMClient({});
  const {
    Policy: { Arn },
  } = await client.send(
    new CreatePolicyCommand({
      PolicyName: NAMES.instancePolicyName,
      PolicyDocument: readFileSync(
        join(RESOURCES_PATH, "instance_policy.json"),
      ),
    }),
  );
  state.instancePolicyArn = Arn;
});
```

```
    }),
    new ScenarioOutput("createdInstancePolicy", (state) =>
      MESSAGES.createdInstancePolicy
        .replace("${INSTANCE_POLICY_NAME}", NAMES.instancePolicyName)
        .replace("${INSTANCE_POLICY_ARN}", state.instancePolicyArn),
    ),
    new ScenarioOutput(
      "creatingInstanceRole",
      MESSAGES.creatingInstanceRole.replace(
        "${INSTANCE_ROLE_NAME}",
        NAMES.instanceRoleName,
      ),
    ),
  ),
  new ScenarioAction("createInstanceRole", () => {
    const client = new IAMClient({});
    return client.send(
      new CreateRoleCommand({
        RoleName: NAMES.instanceRoleName,
        AssumeRolePolicyDocument: readFileSync(
          join(ROOT, "assume-role-policy.json"),
        ),
      }),
    ),
  });
}),
new ScenarioOutput(
  "createdInstanceRole",
  MESSAGES.createdInstanceRole.replace(
    "${INSTANCE_ROLE_NAME}",
    NAMES.instanceRoleName,
  ),
),
new ScenarioOutput(
  "attachingPolicyToRole",
  MESSAGES.attachingPolicyToRole
    .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName)
    .replace("${INSTANCE_POLICY_NAME}", NAMES.instancePolicyName),
),
new ScenarioAction("attachPolicyToRole", async (state) => {
  const client = new IAMClient({});
  await client.send(
    new AttachRolePolicyCommand({
      RoleName: NAMES.instanceRoleName,
      PolicyArn: state.instancePolicyArn,
    }),
  ),
});
```

```
);
}),
new ScenarioOutput(
  "attachedPolicyToRole",
  MESSAGES.attachedPolicyToRole
    .replace("${INSTANCE_POLICY_NAME}", NAMES.instancePolicyName)
    .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName),
),
new ScenarioOutput(
  "creatingInstanceProfile",
  MESSAGES.creatingInstanceProfile.replace(
    "${INSTANCE_PROFILE_NAME}",
    NAMES.instanceProfileName,
  ),
),
new ScenarioAction("createInstanceProfile", async (state) => {
  const client = new IAMClient({});
  const {
    InstanceProfile: { Arn },
  } = await client.send(
    new CreateInstanceProfileCommand({
      InstanceProfileName: NAMES.instanceProfileName,
    }),
  );
  state.instanceProfileArn = Arn;

  await waitUntilInstanceProfileExists(
    { client },
    { InstanceProfileName: NAMES.instanceProfileName },
  );
}),
new ScenarioOutput("createdInstanceProfile", (state) =>
  MESSAGES.createdInstanceProfile
    .replace("${INSTANCE_PROFILE_NAME}", NAMES.instanceProfileName)
    .replace("${INSTANCE_PROFILE_ARN}", state.instanceProfileArn),
),
new ScenarioOutput(
  "addingRoleToInstanceProfile",
  MESSAGES.addingRoleToInstanceProfile
    .replace("${INSTANCE_PROFILE_NAME}", NAMES.instanceProfileName)
    .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName),
),
new ScenarioAction("addRoleToInstanceProfile", () => {
  const client = new IAMClient({});
```

```
return client.send(
  new AddRoleToInstanceProfileCommand({
    RoleName: NAMES.instanceRoleName,
    InstanceProfileName: NAMES.instanceProfileName,
  }),
);
}),
new ScenarioOutput(
  "addedRoleToInstanceProfile",
  MESSAGES.addedRoleToInstanceProfile
    .replace("${INSTANCE_PROFILE_NAME}", NAMES.instanceProfileName)
    .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName),
),
...initParamsSteps,
new ScenarioOutput("creatingLaunchTemplate", MESSAGES.creatingLaunchTemplate),
new ScenarioAction("createLaunchTemplate", async () => {
  // snippet-start:[javascript.v3.wkflw.resilient.CreateLaunchTemplate]
  const ssmClient = new SSMClient({});
  const { Parameter } = await ssmClient.send(
    new GetParameterCommand({
      Name: "/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2",
    }),
  );
  const ec2Client = new EC2Client({});
  await ec2Client.send(
    new CreateLaunchTemplateCommand({
      LaunchTemplateName: NAMES.launchTemplateName,
      LaunchTemplateData: {
        InstanceType: "t3.micro",
        ImageId: Parameter.Value,
        IamInstanceProfile: { Name: NAMES.instanceProfileName },
        UserData: readFileSync(
          join(RESOURCES_PATH, "server_startup_script.sh"),
        ).toString("base64"),
        KeyName: NAMES.keyPairName,
      },
    }),
  );
  // snippet-end:[javascript.v3.wkflw.resilient.CreateLaunchTemplate]
});
}),
new ScenarioOutput(
  "createdLaunchTemplate",
  MESSAGES.createdLaunchTemplate.replace(
    "${LAUNCH_TEMPLATE_NAME}",
```

```

    NAMES.launchTemplateName,
  ),
),
new ScenarioOutput(
  "creatingAutoScalingGroup",
  MESSAGES.creatingAutoScalingGroup.replace(
    "${AUTO_SCALING_GROUP_NAME}",
    NAMES.autoScalingGroupName,
  ),
),
new ScenarioAction("createAutoScalingGroup", async (state) => {
  const ec2Client = new EC2Client({});
  const { AvailabilityZones } = await ec2Client.send(
    new DescribeAvailabilityZonesCommand({}),
  );
  state.availabilityZoneNames = AvailabilityZones.map((az) => az.ZoneName);
  const autoScalingClient = new AutoScalingClient({});
  await retry({ intervalInMs: 1000, maxRetries: 30 }, () =>
    autoScalingClient.send(
      new CreateAutoScalingGroupCommand({
        AvailabilityZones: state.availabilityZoneNames,
        AutoScalingGroupName: NAMES.autoScalingGroupName,
        LaunchTemplate: {
          LaunchTemplateName: NAMES.launchTemplateName,
          Version: "$Default",
        },
        MinSize: 3,
        MaxSize: 3,
      }),
    ),
  );
}),
new ScenarioOutput(
  "createdAutoScalingGroup",
  /**
   * @param {{ availabilityZoneNames: string[] }} state
   */
  (state) =>
    MESSAGES.createdAutoScalingGroup
      .replace("${AUTO_SCALING_GROUP_NAME}", NAMES.autoScalingGroupName)
      .replace(
        "${AVAILABILITY_ZONE_NAMES}",
        state.availabilityZoneNames.join(", "),
      ),
),

```

```

    ),
    new ScenarioInput("confirmContinue", MESSAGES.confirmContinue, {
      type: "confirm",
    }),
    new ScenarioOutput("loadBalancer", MESSAGES.loadBalancer),
    new ScenarioOutput("gettingVpc", MESSAGES.gettingVpc),
    new ScenarioAction("getVpc", async (state) => {
      // snippet-start:[javascript.v3.wkflw.resilient.DescribeVpcs]
      const client = new EC2Client({});
      const { Vpcs } = await client.send(
        new DescribeVpcsCommand({
          Filters: [{ Name: "is-default", Values: ["true"] }],
        }),
      );
      // snippet-end:[javascript.v3.wkflw.resilient.DescribeVpcs]
      state.defaultVpc = Vpcs[0].VpcId;
    }),
    new ScenarioOutput("gotVpc", (state) =>
      MESSAGES.gotVpc.replace("${VPC_ID}", state.defaultVpc),
    ),
    new ScenarioOutput("gettingSubnets", MESSAGES.gettingSubnets),
    new ScenarioAction("getSubnets", async (state) => {
      // snippet-start:[javascript.v3.wkflw.resilient.DescribeSubnets]
      const client = new EC2Client({});
      const { Subnets } = await client.send(
        new DescribeSubnetsCommand({
          Filters: [
            { Name: "vpc-id", Values: [state.defaultVpc] },
            { Name: "availability-zone", Values: state.availabilityZoneNames },
            { Name: "default-for-az", Values: ["true"] },
          ],
        }),
      );
      // snippet-end:[javascript.v3.wkflw.resilient.DescribeSubnets]
      state.subnets = Subnets.map((subnet) => subnet.SubnetId);
    }),
    new ScenarioOutput(
      "gotSubnets",
      /**
       * @param {{ subnets: string[] }} state
       */
      (state) =>
        MESSAGES.gotSubnets.replace("${SUBNETS}", state.subnets.join(", ")),
    ),
  ),

```

```
new ScenarioOutput(
  "creatingLoadBalancerTargetGroup",
  MESSAGES.creatingLoadBalancerTargetGroup.replace(
    "${TARGET_GROUP_NAME}",
    NAMES.loadBalancerTargetGroupName,
  ),
),
new ScenarioAction("createLoadBalancerTargetGroup", async (state) => {
  // snippet-start:[javascript.v3.wkflw.resilient.CreateTargetGroup]
  const client = new ElasticLoadBalancingV2Client({});
  const { TargetGroups } = await client.send(
    new CreateTargetGroupCommand({
      Name: NAMES.loadBalancerTargetGroupName,
      Protocol: "HTTP",
      Port: 80,
      HealthCheckPath: "/healthcheck",
      HealthCheckIntervalSeconds: 10,
      HealthCheckTimeoutSeconds: 5,
      HealthyThresholdCount: 2,
      UnhealthyThresholdCount: 2,
      VpcId: state.defaultVpc,
    }),
  );
  // snippet-end:[javascript.v3.wkflw.resilient.CreateTargetGroup]
  const targetGroup = TargetGroups[0];
  state.targetGroupArn = targetGroup.TargetGroupArn;
  state.targetGroupProtocol = targetGroup.Protocol;
  state.targetGroupPort = targetGroup.Port;
}),
new ScenarioOutput(
  "createdLoadBalancerTargetGroup",
  MESSAGES.createdLoadBalancerTargetGroup.replace(
    "${TARGET_GROUP_NAME}",
    NAMES.loadBalancerTargetGroupName,
  ),
),
new ScenarioOutput(
  "creatingLoadBalancer",
  MESSAGES.creatingLoadBalancer.replace("${LB_NAME}", NAMES.loadBalancerName),
),
new ScenarioAction("createLoadBalancer", async (state) => {
  // snippet-start:[javascript.v3.wkflw.resilient.CreateLoadBalancer]
  const client = new ElasticLoadBalancingV2Client({});
  const { LoadBalancers } = await client.send(
```



```

    new CreateLoadBalancerCommand({
      Name: NAMES.loadBalancerName,
      Subnets: state.subnets,
    }),
  );
state.loadBalancerDns = LoadBalancers[0].DNSName;
state.loadBalancerArn = LoadBalancers[0].LoadBalancerArn;
await waitUntilLoadBalancerAvailable(
  { client },
  { Names: [NAMES.loadBalancerName] },
);
// snippet-end:[javascript.v3.wkflw.resilient.CreateLoadBalancer]
}),
new ScenarioOutput("createdLoadBalancer", (state) =>
  MESSAGES.createdLoadBalancer
    .replace("${LB_NAME}", NAMES.loadBalancerName)
    .replace("${DNS_NAME}", state.loadBalancerDns),
),
new ScenarioOutput(
  "creatingListener",
  MESSAGES.creatingLoadBalancerListener
    .replace("${LB_NAME}", NAMES.loadBalancerName)
    .replace("${TARGET_GROUP_NAME}", NAMES.loadBalancerTargetGroupName),
),
new ScenarioAction("createListener", async (state) => {
  // snippet-start:[javascript.v3.wkflw.resilient.CreateListener]
  const client = new ElasticLoadBalancingV2Client({});
  const { Listeners } = await client.send(
    new CreateListenerCommand({
      LoadBalancerArn: state.loadBalancerArn,
      Protocol: state.targetGroupProtocol,
      Port: state.targetGroupPort,
      DefaultActions: [
        { Type: "forward", TargetGroupArn: state.targetGroupArn },
      ],
    }),
  );
  // snippet-end:[javascript.v3.wkflw.resilient.CreateListener]
  const listener = Listeners[0];
  state.loadBalancerListenerArn = listener.ListenerArn;
}),
new ScenarioOutput("createdListener", (state) =>
  MESSAGES.createdLoadBalancerListener.replace(
    "${LB_LISTENER_ARN}",

```

```

    state.loadBalancerListenerArn,
  ),
),
new ScenarioOutput(
  "attachingLoadBalancerTargetGroup",
  MESSAGES.attachingLoadBalancerTargetGroup
    .replace("${TARGET_GROUP_NAME}", NAMES.loadBalancerTargetGroupName)
    .replace("${AUTO_SCALING_GROUP_NAME}", NAMES.autoScalingGroupName),
),
new ScenarioAction("attachLoadBalancerTargetGroup", async (state) => {
  // snippet-start:[javascript.v3.wkflw.resilient.AttachTargetGroup]
  const client = new AutoScalingClient({});
  await client.send(
    new AttachLoadBalancerTargetGroupsCommand({
      AutoScalingGroupName: NAMES.autoScalingGroupName,
      TargetGroupARNs: [state.targetGroupArn],
    }),
  );
  // snippet-end:[javascript.v3.wkflw.resilient.AttachTargetGroup]
}),
new ScenarioOutput(
  "attachedLoadBalancerTargetGroup",
  MESSAGES.attachedLoadBalancerTargetGroup,
),
new ScenarioOutput("verifyingInboundPort", MESSAGES.verifyingInboundPort),
new ScenarioAction(
  "verifyInboundPort",
  /**
   *
   * @param {{ defaultSecurityGroup: import('@aws-sdk/client-ec2').SecurityGroup}} state
   */
  async (state) => {
    const client = new EC2Client({});
    const { SecurityGroups } = await client.send(
      new DescribeSecurityGroupsCommand({
        Filters: [{ Name: "group-name", Values: ["default"] }],
      }),
    );
    if (!SecurityGroups) {
      state.verifyInboundPortError = new Error(MESSAGES.noSecurityGroups);
    }
    state.defaultSecurityGroup = SecurityGroups[0];
  }
);

```

```
/**
 * @type {string}
 */
const ipResponse = (await axios.get("http://checkip.amazonaws.com")).data;
state.myIp = ipResponse.trim();
const myIpRules = state.defaultSecurityGroup.IpPermissions.filter(
  ({ IpRanges }) =>
    IpRanges.some(
      ({ CidrIp }) =>
        CidrIp.startsWith(state.myIp) || CidrIp === "0.0.0.0/0",
    ),
)
  .filter(({ IpProtocol }) => IpProtocol === "tcp")
  .filter(({ FromPort }) => FromPort === 80);

state.myIpRules = myIpRules;
},
),
new ScenarioOutput(
  "verifiedInboundPort",
  /**
   * @param {{ myIpRules: any[] }} state
   */
  (state) => {
    if (state.myIpRules.length > 0) {
      return MESSAGES.foundIpRules.replace(
        "${IP_RULES}",
        JSON.stringify(state.myIpRules, null, 2),
      );
    } else {
      return MESSAGES.noIpRules;
    }
  },
),
new ScenarioInput(
  "shouldAddInboundRule",
  /**
   * @param {{ myIpRules: any[] }} state
   */
  (state) => {
    if (state.myIpRules.length > 0) {
      return false;
    } else {
      return MESSAGES.noIpRules;
    }
  },
),

```

```

    }
  },
  { type: "confirm" },
),
new ScenarioAction(
  "addInboundRule",
  /**
   * @param {{ defaultSecurityGroup: import('@aws-sdk/client-
ec2').SecurityGroup }} state
   */
  async (state) => {
    if (!state.shouldAddInboundRule) {
      return;
    }

    const client = new EC2Client({});
    await client.send(
      new AuthorizeSecurityGroupIngressCommand({
        GroupId: state.defaultSecurityGroup.GroupId,
        CidrIp: `${state.myIp}/32`,
        FromPort: 80,
        ToPort: 80,
        IpProtocol: "tcp",
      }),
    );
  },
),
new ScenarioOutput("addedInboundRule", (state) => {
  if (state.shouldAddInboundRule) {
    return MESSAGES.addedInboundRule.replace("${IP_ADDRESS}", state.myIp);
  } else {
    return false;
  }
}),
new ScenarioOutput("verifyingEndpoint", (state) =>
  MESSAGES.verifyingEndpoint.replace("${DNS_NAME}", state.loadBalancerDns),
),
new ScenarioAction("verifyEndpoint", async (state) => {
  try {
    const response = await retry({ intervalInMs: 2000, maxRetries: 30 }, () =>
      axios.get(`http://${state.loadBalancerDns}`),
    );
    state.endpointResponse = JSON.stringify(response.data, null, 2);
  } catch (e) {

```

```
        state.verifyEndpointError = e;
    }
  })),
  new ScenarioOutput("verifiedEndpoint", (state) => {
    if (state.verifyEndpointError) {
      console.error(state.verifyEndpointError);
    } else {
      return MESSAGES.verifiedEndpoint.replace(
        "${ENDPOINT_RESPONSE}",
        state.endpointResponse,
      );
    }
  })),
];
```

Cree los pasos para ejecutar la demostración.

```
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: Apache-2.0
import { readFileSync } from "node:fs";
import { join } from "node:path";

import axios from "axios";

import {
  DescribeTargetGroupsCommand,
  DescribeTargetHealthCommand,
  ElasticLoadBalancingV2Client,
} from "@aws-sdk/client-elastic-load-balancing-v2";
import {
  DescribeInstanceInformationCommand,
  PutParameterCommand,
  SSMClient,
  SendCommandCommand,
} from "@aws-sdk/client-ssm";
import {
  IAMClient,
  CreatePolicyCommand,
  CreateRoleCommand,
  AttachRolePolicyCommand,
  CreateInstanceProfileCommand,
  AddRoleToInstanceProfileCommand,
```

```
    waitUntilInstanceProfileExists,
  } from "@aws-sdk/client-iam";
import {
  AutoScalingClient,
  DescribeAutoScalingGroupsCommand,
  TerminateInstanceInAutoScalingGroupCommand,
} from "@aws-sdk/client-auto-scaling";
import {
  DescribeIamInstanceProfileAssociationsCommand,
  EC2Client,
  RebootInstancesCommand,
  ReplaceIamInstanceProfileAssociationCommand,
} from "@aws-sdk/client-ec2";

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

import { MESSAGES, NAMES, RESOURCES_PATH } from "./constants.js";
import { findLoadBalancer } from "./shared.js";

const getRecommendation = new ScenarioAction(
  "getRecommendation",
  async (state) => {
    const loadBalancer = await findLoadBalancer(NAMES.loadBalancerName);
    if (loadBalancer) {
      state.loadBalancerDnsName = loadBalancer.DNSName;
      try {
        state.recommendation = (
          await axios.get(`http://${state.loadBalancerDnsName}`)
        ).data;
      } catch (e) {
        state.recommendation = e instanceof Error ? e.message : e;
      }
    } else {
      throw new Error(MESSAGES.demoFindLoadBalancerError);
    }
  },
);

const getRecommendationResult = new ScenarioOutput(
```

```

"getRecommendationResult",
(state) =>
  `Recommendation:\n${JSON.stringify(state.recommendation, null, 2)}`,
  { preformatted: true },
);

const getHealthCheck = new ScenarioAction("getHealthCheck", async (state) => {
  // snippet-start:[javascript.v3.wkflw.resilient.DescribeTargetGroups]
  const client = new ElasticLoadBalancingV2Client({});
  const { TargetGroups } = await client.send(
    new DescribeTargetGroupsCommand({
      Names: [NAMES.loadBalancerTargetGroupName],
    }),
  );
  // snippet-end:[javascript.v3.wkflw.resilient.DescribeTargetGroups]

  // snippet-start:[javascript.v3.wkflw.resilient.DescribeTargetHealth]
  const { TargetHealthDescriptions } = await client.send(
    new DescribeTargetHealthCommand({
      TargetGroupArn: TargetGroups[0].TargetGroupArn,
    }),
  );
  // snippet-end:[javascript.v3.wkflw.resilient.DescribeTargetHealth]
  state.targetHealthDescriptions = TargetHealthDescriptions;
});

const getHealthCheckResult = new ScenarioOutput(
  "getHealthCheckResult",
  /**
   * @param {{ targetHealthDescriptions: import('@aws-sdk/client-elastic-load-balancing-v2').TargetHealthDescription[][]} state
   */
  (state) => {
    const status = state.targetHealthDescriptions
      .map((th) => `${th.Target.Id}: ${th.TargetHealth.State}`)
      .join("\n");
    return `Health check:\n${status}`;
  },
  { preformatted: true },
);

const loadBalancerLoop = new ScenarioAction(
  "loadBalancerLoop",
  getRecommendation.action,

```

```
{
  whileConfig: {
    whileFn: ({ loadBalancerCheck }) => loadBalancerCheck,
    input: new ScenarioInput(
      "loadBalancerCheck",
      MESSAGES.demoLoadBalancerCheck,
      {
        type: "confirm",
      },
    ),
    output: getRecommendationResult,
  },
},
);

const healthCheckLoop = new ScenarioAction(
  "healthCheckLoop",
  getHealthCheck.action,
  {
    whileConfig: {
      whileFn: ({ healthCheck }) => healthCheck,
      input: new ScenarioInput("healthCheck", MESSAGES.demoHealthCheck, {
        type: "confirm",
      }),
      output: getHealthCheckResult,
    },
  },
);

const statusSteps = [
  getRecommendation,
  getRecommendationResult,
  getHealthCheck,
  getHealthCheckResult,
];

/**
 * @type {import('@aws-doc-sdk-examples/lib/scenario.js').Step[]}
 */
export const demoSteps = [
  new ScenarioOutput("header", MESSAGES.demoHeader, { header: true }),
  new ScenarioOutput("sanityCheck", MESSAGES.demoSanityCheck),
  ...statusSteps,
  new ScenarioInput(
```



```
"brokenDependencyConfirmation",
MESSAGES.demoBrokenDependencyConfirmation,
{ type: "confirm" },
),
new ScenarioAction("brokenDependency", async (state) => {
  if (!state.brokenDependencyConfirmation) {
    process.exit();
  } else {
    const client = new SSMClient({});
    state.badTableName = `fake-table-${Date.now()}`;
    await client.send(
      new PutParameterCommand({
        Name: NAMES.ssmTableNameKey,
        Value: state.badTableName,
        Overwrite: true,
        Type: "String",
      }),
    );
  }
}),
new ScenarioOutput("testBrokenDependency", (state) =>
  MESSAGES.demoTestBrokenDependency.replace(
    "${TABLE_NAME}",
    state.badTableName,
  ),
),
...statusSteps,
new ScenarioInput(
  "staticResponseConfirmation",
  MESSAGES.demoStaticResponseConfirmation,
  { type: "confirm" },
),
new ScenarioAction("staticResponse", async (state) => {
  if (!state.staticResponseConfirmation) {
    process.exit();
  } else {
    const client = new SSMClient({});
    await client.send(
      new PutParameterCommand({
        Name: NAMES.ssmFailureResponseKey,
        Value: "static",
        Overwrite: true,
        Type: "String",
      }),
    ),
  }
}),
```

```

    );
  }
}),
new ScenarioOutput("testStaticResponse", MESSAGES.demoTestStaticResponse),
...statusSteps,
new ScenarioInput(
  "badCredentialsConfirmation",
  MESSAGES.demoBadCredentialsConfirmation,
  { type: "confirm" },
),
new ScenarioAction("badCredentialsExit", (state) => {
  if (!state.badCredentialsConfirmation) {
    process.exit();
  }
}),
new ScenarioAction("fixDynamoDBName", async () => {
  const client = new SSMClient({});
  await client.send(
    new PutParameterCommand({
      Name: NAMES.ssmTableNameKey,
      Value: NAMES.tableName,
      Overwrite: true,
      Type: "String",
    }),
  ),
),
),
new ScenarioAction(
  "badCredentials",
  /**
   * @param {{ targetInstance: import('@aws-sdk/client-auto-
scaling').Instance }} state
   */
  async (state) => {
    await createSsmOnlyInstanceProfile();
    const autoScalingClient = new AutoScalingClient({});
    const { AutoScalingGroups } = await autoScalingClient.send(
      new DescribeAutoScalingGroupsCommand({
        AutoScalingGroupNames: [NAMES.autoScalingGroupName],
      }),
    ),
  );
  state.targetInstance = AutoScalingGroups[0].Instances[0];
  // snippet-start:
[javascript.v3.wkflw.resilient.DescribeIamInstanceProfileAssociations]
  const ec2Client = new EC2Client({});

```

```

const { IamInstanceProfileAssociations } = await ec2Client.send(
  new DescribeIamInstanceProfileAssociationsCommand({
    Filters: [
      { Name: "instance-id", Values: [state.targetInstance.InstanceId] },
    ],
  }),
);
// snippet-end:
[javascript.v3.wkflw.resilient.DescribeIamInstanceProfileAssociations]
state.instanceProfileAssociationId =
  IamInstanceProfileAssociations[0].AssociationId;
// snippet-start:
[javascript.v3.wkflw.resilient.ReplaceIamInstanceProfileAssociation]
await retry({ intervalInMs: 1000, maxRetries: 30 }, () =>
  ec2Client.send(
    new ReplaceIamInstanceProfileAssociationCommand({
      AssociationId: state.instanceProfileAssociationId,
      IamInstanceProfile: { Name: NAMES.ssmOnlyInstanceProfileName },
    }),
  ),
);
// snippet-end:
[javascript.v3.wkflw.resilient.ReplaceIamInstanceProfileAssociation]

await ec2Client.send(
  new RebootInstancesCommand({
    InstanceIds: [state.targetInstance.InstanceId],
  }),
);

const ssmClient = new SSMClient({});
await retry({ intervalInMs: 20000, maxRetries: 15 }, async () => {
  const { InstanceInformationList } = await ssmClient.send(
    new DescribeInstanceInformationCommand({}),
  );

  const instance = InstanceInformationList.find(
    (info) => info.InstanceId === state.targetInstance.InstanceId,
  );

  if (!instance) {
    throw new Error("Instance not found.");
  }
});

```

```

    await ssmClient.send(
      new SendCommandCommand({
        InstanceIds: [state.targetInstance.InstanceId],
        DocumentName: "AWS-RunShellScript",
        Parameters: { commands: ["cd / && sudo python3 server.py 80"] },
      }),
    );
  },
),
new ScenarioOutput(
  "testBadCredentials",
  /**
   * @param {{ targetInstance: import('@aws-sdk/client-
   ssm').InstanceInformation}} state
   */
  (state) =>
    MESSAGES.demoTestBadCredentials.replace(
      "${INSTANCE_ID}",
      state.targetInstance.InstanceId,
    ),
),
loadBalancerLoop,
new ScenarioInput(
  "deepHealthCheckConfirmation",
  MESSAGES.demoDeepHealthCheckConfirmation,
  { type: "confirm" },
),
new ScenarioAction("deepHealthCheckExit", (state) => {
  if (!state.deepHealthCheckConfirmation) {
    process.exit();
  }
}),
new ScenarioAction("deepHealthCheck", async () => {
  const client = new SSMClient({});
  await client.send(
    new PutParameterCommand({
      Name: NAMES.ssmHealthCheckKey,
      Value: "deep",
      Overwrite: true,
      Type: "String",
    }),
  );
}),
}),

```

```

new ScenarioOutput("testDeepHealthCheck", MESSAGES.demoTestDeepHealthCheck),
healthCheckLoop,
loadBalancerLoop,
new ScenarioInput(
  "killInstanceConfirmation",
  /**
   * @param {{ targetInstance: import('@aws-sdk/client-
ssm').InstanceInformation }} state
   */
  (state) =>
    MESSAGES.demoKillInstanceConfirmation.replace(
      "${INSTANCE_ID}",
      state.targetInstance.InstanceId,
    ),
  { type: "confirm" },
),
new ScenarioAction("killInstanceExit", (state) => {
  if (!state.killInstanceConfirmation) {
    process.exit();
  }
}),
new ScenarioAction(
  "killInstance",
  /**
   * @param {{ targetInstance: import('@aws-sdk/client-
ssm').InstanceInformation }} state
   */
  async (state) => {
    const client = new AutoScalingClient({});
    await client.send(
      new TerminateInstanceInAutoScalingGroupCommand({
        InstanceId: state.targetInstance.InstanceId,
        ShouldDecrementDesiredCapacity: false,
      }),
    );
  },
),
new ScenarioOutput("testKillInstance", MESSAGES.demoTestKillInstance),
healthCheckLoop,
loadBalancerLoop,
new ScenarioInput("failOpenConfirmation", MESSAGES.demoFailOpenConfirmation, {
  type: "confirm",
}),
new ScenarioAction("failOpenExit", (state) => {

```

```
    if (!state.failOpenConfirmation) {
      process.exit();
    }
  })),
  new ScenarioAction("failOpen", () => {
    const client = new SSMClient({});
    return client.send(
      new PutParameterCommand({
        Name: NAMES.ssmTableNameKey,
        Value: `fake-table-${Date.now()}`,
        Overwrite: true,
        Type: "String",
      }),
    );
  })),
  new ScenarioOutput("testFailOpen", MESSAGES.demoFailOpenTest),
  healthCheckLoop,
  loadBalancerLoop,
  new ScenarioInput(
    "resetTableConfirmation",
    MESSAGES.demoResetTableConfirmation,
    { type: "confirm" },
  ),
  new ScenarioAction("resetTableExit", (state) => {
    if (!state.resetTableConfirmation) {
      process.exit();
    }
  })),
  new ScenarioAction("resetTable", async () => {
    const client = new SSMClient({});
    await client.send(
      new PutParameterCommand({
        Name: NAMES.ssmTableNameKey,
        Value: NAMES.tableName,
        Overwrite: true,
        Type: "String",
      }),
    );
  })),
  new ScenarioOutput("testResetTable", MESSAGES.demoTestResetTable),
  healthCheckLoop,
  loadBalancerLoop,
];
```

```
async function createSsmOnlyInstanceProfile() {
  const iamClient = new IAMClient({});
  const { Policy } = await iamClient.send(
    new CreatePolicyCommand({
      PolicyName: NAMES.ssmOnlyPolicyName,
      PolicyDocument: readFileSync(
        join(RESOURCES_PATH, "ssm_only_policy.json"),
      ),
    }),
  );
  await iamClient.send(
    new CreateRoleCommand({
      RoleName: NAMES.ssmOnlyRoleName,
      AssumeRolePolicyDocument: JSON.stringify({
        Version: "2012-10-17",
        Statement: [
          {
            Effect: "Allow",
            Principal: { Service: "ec2.amazonaws.com" },
            Action: "sts:AssumeRole",
          },
        ],
      }),
    ),
  );
  await iamClient.send(
    new AttachRolePolicyCommand({
      RoleName: NAMES.ssmOnlyRoleName,
      PolicyArn: Policy.Arn,
    }),
  );
  await iamClient.send(
    new AttachRolePolicyCommand({
      RoleName: NAMES.ssmOnlyRoleName,
      PolicyArn: "arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore",
    }),
  );
  // snippet-start:[javascript.v3.wkflw.resilient.CreateInstanceProfile]
  const { InstanceProfile } = await iamClient.send(
    new CreateInstanceProfileCommand({
      InstanceProfileName: NAMES.ssmOnlyInstanceProfileName,
    }),
  );
  await waitUntilInstanceProfileExists(
```

```

    { client: iamClient },
    { InstanceProfileName: NAMES.ssmOnlyInstanceProfileName },
  );
// snippet-end:[javascript.v3.wkflw.resilient.CreateInstanceProfile]
await iamClient.send(
  new AddRoleToInstanceProfileCommand({
    InstanceProfileName: NAMES.ssmOnlyInstanceProfileName,
    RoleName: NAMES.ssmOnlyRoleName,
  })),
);

return InstanceProfile;
}

```

Cree los pasos para destruir todos los recursos.

```

// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: Apache-2.0
import { unlinkSync } from "node:fs";

import { DynamoDBClient, DeleteTableCommand } from "@aws-sdk/client-dynamodb";
import {
  EC2Client,
  DeleteKeyPairCommand,
  DeleteLaunchTemplateCommand,
} from "@aws-sdk/client-ec2";
import {
  IAMClient,
  DeleteInstanceProfileCommand,
  RemoveRoleFromInstanceProfileCommand,
  DeletePolicyCommand,
  DeleteRoleCommand,
  DetachRolePolicyCommand,
  paginateListPolicies,
} from "@aws-sdk/client-iam";
import {
  AutoScalingClient,
  DeleteAutoScalingGroupCommand,
  TerminateInstanceInAutoScalingGroupCommand,
  UpdateAutoScalingGroupCommand,
  paginateDescribeAutoScalingGroups,
} from "@aws-sdk/client-auto-scaling";

```



```
import {
  DeleteLoadBalancerCommand,
  DeleteTargetGroupCommand,
  DescribeTargetGroupsCommand,
  ElasticLoadBalancingV2Client,
} from "@aws-sdk/client-elastic-load-balancing-v2";

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

import { MESSAGES, NAMES } from "./constants.js";
import { findLoadBalancer } from "./shared.js";

/**
 * @type {import('@aws-doc-sdk-examples/lib/scenario.js').Step[]}
 */
export const destroySteps = [
  new ScenarioInput("destroy", MESSAGES.destroy, { type: "confirm" }),
  new ScenarioAction(
    "abort",
    (state) => state.destroy === false && process.exit(),
  ),
  new ScenarioAction("deleteTable", async (c) => {
    try {
      const client = new DynamoDBClient({});
      await client.send(new DeleteTableCommand({ TableName: NAMES.tableName }));
    } catch (e) {
      c.deleteTableError = e;
    }
  }),
  new ScenarioOutput("deleteTableResult", (state) => {
    if (state.deleteTableError) {
      console.error(state.deleteTableError);
      return MESSAGES.deleteTableError.replace(
        "${TABLE_NAME}",
        NAMES.tableName,
      );
    } else {
      return MESSAGES.deletedTable.replace("${TABLE_NAME}", NAMES.tableName);
    }
  })
];
```

```
    }),
    new ScenarioAction("deleteKeyPair", async (state) => {
      try {
        const client = new EC2Client({});
        await client.send(
          new DeleteKeyPairCommand({ KeyName: NAMES.keyPairName }),
        );
        unlinkSync(`${NAMES.keyPairName}.pem`);
      } catch (e) {
        state.deleteKeyPairError = e;
      }
    }),
    new ScenarioOutput("deleteKeyPairResult", (state) => {
      if (state.deleteKeyPairError) {
        console.error(state.deleteKeyPairError);
        return MESSAGES.deleteKeyPairError.replace(
          "${KEY_PAIR_NAME}",
          NAMES.keyPairName,
        );
      } else {
        return MESSAGES.deletedKeyPair.replace(
          "${KEY_PAIR_NAME}",
          NAMES.keyPairName,
        );
      }
    }),
    new ScenarioAction("detachPolicyFromRole", async (state) => {
      try {
        const client = new IAMClient({});
        const policy = await findPolicy(NAMES.instancePolicyName);

        if (!policy) {
          state.detachPolicyFromRoleError = new Error(
            `Policy ${NAMES.instancePolicyName} not found.`,
          );
        } else {
          await client.send(
            new DetachRolePolicyCommand({
              RoleName: NAMES.instanceRoleName,
              PolicyArn: policy.Arn,
            }),
          );
        }
      } catch (e) {
```

```
    state.detachPolicyFromRoleError = e;
  }
}),
new ScenarioOutput("detachedPolicyFromRole", (state) => {
  if (state.detachPolicyFromRoleError) {
    console.error(state.detachPolicyFromRoleError);
    return MESSAGES.detachPolicyFromRoleError
      .replace("${INSTANCE_POLICY_NAME}", NAMES.instancePolicyName)
      .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName);
  } else {
    return MESSAGES.detachedPolicyFromRole
      .replace("${INSTANCE_POLICY_NAME}", NAMES.instancePolicyName)
      .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName);
  }
}),
new ScenarioAction("deleteInstancePolicy", async (state) => {
  const client = new IAMClient({});
  const policy = await findPolicy(NAMES.instancePolicyName);

  if (!policy) {
    state.deletePolicyError = new Error(
      `Policy ${NAMES.instancePolicyName} not found.`
    );
  } else {
    return client.send(
      new DeletePolicyCommand({
        PolicyArn: policy.Arn,
      })
    );
  }
}),
new ScenarioOutput("deletePolicyResult", (state) => {
  if (state.deletePolicyError) {
    console.error(state.deletePolicyError);
    return MESSAGES.deletePolicyError.replace(
      "${INSTANCE_POLICY_NAME}",
      NAMES.instancePolicyName,
    );
  } else {
    return MESSAGES.deletedPolicy.replace(
      "${INSTANCE_POLICY_NAME}",
      NAMES.instancePolicyName,
    );
  }
})
```



```

    );
  } else {
    return MESSAGES.deletedInstanceRole.replace(
      "${INSTANCE_ROLE_NAME}",
      NAMES.instanceRoleName,
    );
  }
}),
new ScenarioAction("deleteInstanceProfile", async (state) => {
  try {
    // snippet-start:[javascript.v3.wkflw.resilient.DeleteInstanceProfile]
    const client = new IAMClient({});
    await client.send(
      new DeleteInstanceProfileCommand({
        InstanceProfileName: NAMES.instanceProfileName,
      }),
    );
    // snippet-end:[javascript.v3.wkflw.resilient.DeleteInstanceProfile]
  } catch (e) {
    state.deleteInstanceProfileError = e;
  }
}),
new ScenarioOutput("deleteInstanceProfileResult", (state) => {
  if (state.deleteInstanceProfileError) {
    console.error(state.deleteInstanceProfileError);
    return MESSAGES.deleteInstanceProfileError.replace(
      "${INSTANCE_PROFILE_NAME}",
      NAMES.instanceProfileName,
    );
  } else {
    return MESSAGES.deletedInstanceProfile.replace(
      "${INSTANCE_PROFILE_NAME}",
      NAMES.instanceProfileName,
    );
  }
}),
new ScenarioAction("deleteAutoScalingGroup", async (state) => {
  try {
    await terminateGroupInstances(NAMES.autoScalingGroupName);
    await retry({ intervalInMs: 60000, maxRetries: 60 }, async () => {
      await deleteAutoScalingGroup(NAMES.autoScalingGroupName);
    });
  } catch (e) {
    state.deleteAutoScalingGroupError = e;
  }
});

```

```
    }
  })),
  new ScenarioOutput("deleteAutoScalingGroupResult", (state) => {
    if (state.deleteAutoScalingGroupError) {
      console.error(state.deleteAutoScalingGroupError);
      return MESSAGES.deleteAutoScalingGroupError.replace(
        "${AUTO_SCALING_GROUP_NAME}",
        NAMES.autoScalingGroupName,
      );
    } else {
      return MESSAGES.deletedAutoScalingGroup.replace(
        "${AUTO_SCALING_GROUP_NAME}",
        NAMES.autoScalingGroupName,
      );
    }
  })),
  new ScenarioAction("deleteLaunchTemplate", async (state) => {
    const client = new EC2Client({});
    try {
      // snippet-start:[javascript.v3.wkflw.resilient.DeleteLaunchTemplate]
      await client.send(
        new DeleteLaunchTemplateCommand({
          LaunchTemplateName: NAMES.launchTemplateName,
        }),
      );
      // snippet-end:[javascript.v3.wkflw.resilient.DeleteLaunchTemplate]
    } catch (e) {
      state.deleteLaunchTemplateError = e;
    }
  })),
  new ScenarioOutput("deleteLaunchTemplateResult", (state) => {
    if (state.deleteLaunchTemplateError) {
      console.error(state.deleteLaunchTemplateError);
      return MESSAGES.deleteLaunchTemplateError.replace(
        "${LAUNCH_TEMPLATE_NAME}",
        NAMES.launchTemplateName,
      );
    } else {
      return MESSAGES.deletedLaunchTemplate.replace(
        "${LAUNCH_TEMPLATE_NAME}",
        NAMES.launchTemplateName,
      );
    }
  })),
}
```

```
new ScenarioAction("deleteLoadBalancer", async (state) => {
  try {
    // snippet-start:[javascript.v3.wkflw.resilient.DeleteLoadBalancer]
    const client = new ElasticLoadBalancingV2Client({});
    const loadBalancer = await findLoadBalancer(NAMES.loadBalancerName);
    await client.send(
      new DeleteLoadBalancerCommand({
        LoadBalancerArn: loadBalancer.LoadBalancerArn,
      }),
    );
    await retry({ intervalInMs: 1000, maxRetries: 60 }, async () => {
      const lb = await findLoadBalancer(NAMES.loadBalancerName);
      if (lb) {
        throw new Error("Load balancer still exists.");
      }
    });
    // snippet-end:[javascript.v3.wkflw.resilient.DeleteLoadBalancer]
  } catch (e) {
    state.deleteLoadBalancerError = e;
  }
}),
new ScenarioOutput("deleteLoadBalancerResult", (state) => {
  if (state.deleteLoadBalancerError) {
    console.error(state.deleteLoadBalancerError);
    return MESSAGES.deleteLoadBalancerError.replace(
      "${LB_NAME}",
      NAMES.loadBalancerName,
    );
  } else {
    return MESSAGES.deletedLoadBalancer.replace(
      "${LB_NAME}",
      NAMES.loadBalancerName,
    );
  }
}),
new ScenarioAction("deleteLoadBalancerTargetGroup", async (state) => {
  // snippet-start:[javascript.v3.wkflw.resilient.DeleteTargetGroup]
  const client = new ElasticLoadBalancingV2Client({});
  try {
    const { TargetGroups } = await client.send(
      new DescribeTargetGroupsCommand({
        Names: [NAMES.loadBalancerTargetGroupName],
      }),
    );
  }
});
```

```

    await retry({ intervalInMs: 1000, maxRetries: 30 }, () =>
      client.send(
        new DeleteTargetGroupCommand({
          TargetGroupArn: TargetGroups[0].TargetGroupArn,
        }),
      ),
    );
  } catch (e) {
    state.deleteLoadBalancerTargetGroupError = e;
  }
  // snippet-end:[javascript.v3.wkflw.resilient.DeleteTargetGroup]
}),
new ScenarioOutput("deleteLoadBalancerTargetGroupResult", (state) => {
  if (state.deleteLoadBalancerTargetGroupError) {
    console.error(state.deleteLoadBalancerTargetGroupError);
    return MESSAGES.deleteLoadBalancerTargetGroupError.replace(
      "${TARGET_GROUP_NAME}",
      NAMES.loadBalancerTargetGroupName,
    );
  } else {
    return MESSAGES.deletedLoadBalancerTargetGroup.replace(
      "${TARGET_GROUP_NAME}",
      NAMES.loadBalancerTargetGroupName,
    );
  }
}),
new ScenarioAction("detachSsmOnlyRoleFromProfile", async (state) => {
  try {
    const client = new IAMClient({});
    await client.send(
      new RemoveRoleFromInstanceProfileCommand({
        InstanceProfileName: NAMES.ssmOnlyInstanceProfileName,
        RoleName: NAMES.ssmOnlyRoleName,
      }),
    );
  } catch (e) {
    state.detachSsmOnlyRoleFromProfileError = e;
  }
}),
new ScenarioOutput("detachSsmOnlyRoleFromProfileResult", (state) => {
  if (state.detachSsmOnlyRoleFromProfileError) {
    console.error(state.detachSsmOnlyRoleFromProfileError);
    return MESSAGES.detachSsmOnlyRoleFromProfileError

```



```

        .replace("${ROLE_NAME}", NAMES.ssmOnlyRoleName)
        .replace("${PROFILE_NAME}", NAMES.ssmOnlyInstanceProfileName);
    } else {
        return MESSAGES.detachedSsmOnlyRoleFromProfile
            .replace("${ROLE_NAME}", NAMES.ssmOnlyRoleName)
            .replace("${PROFILE_NAME}", NAMES.ssmOnlyInstanceProfileName);
    }
}),
new ScenarioAction("detachSsmOnlyCustomRolePolicy", async (state) => {
    try {
        const iamClient = new IAMClient({});
        const ssmOnlyPolicy = await findPolicy(NAMES.ssmOnlyPolicyName);
        await iamClient.send(
            new DetachRolePolicyCommand({
                RoleName: NAMES.ssmOnlyRoleName,
                PolicyArn: ssmOnlyPolicy.Arn,
            })),
        );
    } catch (e) {
        state.detachSsmOnlyCustomRolePolicyError = e;
    }
}),
new ScenarioOutput("detachSsmOnlyCustomRolePolicyResult", (state) => {
    if (state.detachSsmOnlyCustomRolePolicyError) {
        console.error(state.detachSsmOnlyCustomRolePolicyError);
        return MESSAGES.detachSsmOnlyCustomRolePolicyError
            .replace("${ROLE_NAME}", NAMES.ssmOnlyRoleName)
            .replace("${POLICY_NAME}", NAMES.ssmOnlyPolicyName);
    } else {
        return MESSAGES.detachedSsmOnlyCustomRolePolicy
            .replace("${ROLE_NAME}", NAMES.ssmOnlyRoleName)
            .replace("${POLICY_NAME}", NAMES.ssmOnlyPolicyName);
    }
}),
new ScenarioAction("detachSsmOnlyAWSRolePolicy", async (state) => {
    try {
        const iamClient = new IAMClient({});
        await iamClient.send(
            new DetachRolePolicyCommand({
                RoleName: NAMES.ssmOnlyRoleName,
                PolicyArn: "arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore",
            })),
        );
    } catch (e) {

```

```

    state.detachSsmOnlyAWSRolePolicyError = e;
  }
}),
new ScenarioOutput("detachSsmOnlyAWSRolePolicyResult", (state) => {
  if (state.detachSsmOnlyAWSRolePolicyError) {
    console.error(state.detachSsmOnlyAWSRolePolicyError);
    return MESSAGES.detachSsmOnlyAWSRolePolicyError
      .replace("${ROLE_NAME}", NAMES.ssmOnlyRoleName)
      .replace("${POLICY_NAME}", "AmazonSSMManagedInstanceCore");
  } else {
    return MESSAGES.detachedSsmOnlyAWSRolePolicy
      .replace("${ROLE_NAME}", NAMES.ssmOnlyRoleName)
      .replace("${POLICY_NAME}", "AmazonSSMManagedInstanceCore");
  }
}),
new ScenarioAction("deleteSsmOnlyInstanceProfile", async (state) => {
  try {
    const iamClient = new IAMClient({});
    await iamClient.send(
      new DeleteInstanceProfileCommand({
        InstanceProfileName: NAMES.ssmOnlyInstanceProfileName,
      }),
    );
  } catch (e) {
    state.deleteSsmOnlyInstanceProfileError = e;
  }
}),
new ScenarioOutput("deleteSsmOnlyInstanceProfileResult", (state) => {
  if (state.deleteSsmOnlyInstanceProfileError) {
    console.error(state.deleteSsmOnlyInstanceProfileError);
    return MESSAGES.deleteSsmOnlyInstanceProfileError.replace(
      "${INSTANCE_PROFILE_NAME}",
      NAMES.ssmOnlyInstanceProfileName,
    );
  } else {
    return MESSAGES.deletedSsmOnlyInstanceProfile.replace(
      "${INSTANCE_PROFILE_NAME}",
      NAMES.ssmOnlyInstanceProfileName,
    );
  }
}),
new ScenarioAction("deleteSsmOnlyPolicy", async (state) => {
  try {
    const iamClient = new IAMClient({});

```

```
const ssmOnlyPolicy = await findPolicy(NAMES.ssmOnlyPolicyName);
await iamClient.send(
  new DeletePolicyCommand({
    PolicyArn: ssmOnlyPolicy.Arn,
  }),
);
} catch (e) {
  state.deleteSsmOnlyPolicyError = e;
}
}),
new ScenarioOutput("deleteSsmOnlyPolicyResult", (state) => {
  if (state.deleteSsmOnlyPolicyError) {
    console.error(state.deleteSsmOnlyPolicyError);
    return MESSAGES.deleteSsmOnlyPolicyError.replace(
      "${POLICY_NAME}",
      NAMES.ssmOnlyPolicyName,
    );
  } else {
    return MESSAGES.deletedSsmOnlyPolicy.replace(
      "${POLICY_NAME}",
      NAMES.ssmOnlyPolicyName,
    );
  }
}),
new ScenarioAction("deleteSsmOnlyRole", async (state) => {
  try {
    const iamClient = new IAMClient({});
    await iamClient.send(
      new DeleteRoleCommand({
        RoleName: NAMES.ssmOnlyRoleName,
      }),
    );
  } catch (e) {
    state.deleteSsmOnlyRoleError = e;
  }
}),
new ScenarioOutput("deleteSsmOnlyRoleResult", (state) => {
  if (state.deleteSsmOnlyRoleError) {
    console.error(state.deleteSsmOnlyRoleError);
    return MESSAGES.deleteSsmOnlyRoleError.replace(
      "${ROLE_NAME}",
      NAMES.ssmOnlyRoleName,
    );
  } else {
```

```
        return MESSAGES.deletedSsmOnlyRole.replace(
            "${ROLE_NAME}",
            NAMES.ssmOnlyRoleName,
        );
    }
}),
];

/**
 * @param {string} policyName
 */
async function findPolicy(policyName) {
    const client = new IAMClient({});
    const paginatedPolicies = paginateListPolicies({ client }, {});
    for await (const page of paginatedPolicies) {
        const policy = page.Policies.find((p) => p.PolicyName === policyName);
        if (policy) {
            return policy;
        }
    }
}

/**
 * @param {string} groupName
 */
async function deleteAutoScalingGroup(groupName) {
    const client = new AutoScalingClient({});
    try {
        await client.send(
            new DeleteAutoScalingGroupCommand({
                AutoScalingGroupName: groupName,
            }),
        );
    } catch (err) {
        if (!(err instanceof Error)) {
            throw err;
        } else {
            console.log(err.name);
            throw err;
        }
    }
}

/**
```

```
* @param {string} groupName
*/
async function terminateGroupInstances(groupName) {
  const autoScalingClient = new AutoScalingClient({});
  const group = await findAutoScalingGroup(groupName);
  await autoScalingClient.send(
    new UpdateAutoScalingGroupCommand({
      AutoScalingGroupName: group.AutoScalingGroupName,
      MinSize: 0,
    }),
  );
  for (const i of group.Instances) {
    await retry({ intervalInMs: 1000, maxRetries: 30 }, () =>
      autoScalingClient.send(
        new TerminateInstanceInAutoScalingGroupCommand({
          InstanceId: i.InstanceId,
          ShouldDecrementDesiredCapacity: true,
        }),
      ),
    );
  }
}

async function findAutoScalingGroup(groupName) {
  const client = new AutoScalingClient({});
  const paginatedGroups = paginateDescribeAutoScalingGroups({ client }, {});
  for await (const page of paginatedGroups) {
    const group = page.AutoScalingGroups.find(
      (g) => g.AutoScalingGroupName === groupName,
    );
    if (group) {
      return group;
    }
  }
  throw new Error(`Auto scaling group ${groupName} not found.`);
}
```

- Para obtener información sobre la API, consulte los siguientes temas en la Referencia de la API de AWS SDK for JavaScript .
  - [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

Hay más información GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

## Ejecute el escenario interactivo en un símbolo del sistema.

```
class Runner:
    def __init__(
        self, resource_path, recommendation, autoscaler, loadbalancer,
        param_helper
    ):
        self.resource_path = resource_path
        self.recommendation = recommendation
        self.autoscaler = autoscaler
        self.loadbalancer = loadbalancer
        self.param_helper = param_helper
        self.protocol = "HTTP"
        self.port = 80
        self.ssh_port = 22

    def deploy(self):
        recommendations_path = f"{self.resource_path}/recommendations.json"
        startup_script = f"{self.resource_path}/server_startup_script.sh"
        instance_policy = f"{self.resource_path}/instance_policy.json"

        print(
            "\nFor this demo, we'll use the AWS SDK for Python (Boto3) to create
            several AWS resources\n"
            "to set up a load-balanced web service endpoint and explore some ways
            to make it resilient\n"
            "against various kinds of failures.\n\n"
            "Some of the resources create by this demo are:\n"
        )
        print(
            "\t* A DynamoDB table that the web service depends on to provide
            book, movie, and song recommendations."
        )
        print(
            "\t* An EC2 launch template that defines EC2 instances that each
            contain a Python web server."
        )
        print(
            "\t* An EC2 Auto Scaling group that manages EC2 instances across
            several Availability Zones."
        )
        print(
            "\t* An Elastic Load Balancing (ELB) load balancer that targets the
            Auto Scaling group to distribute requests."
```

```
)
print("-" * 88)
q.ask("Press Enter when you're ready to start deploying resources.")

print(
    f"Creating and populating a DynamoDB table named
'{self.recommendation.table_name}'."
)
self.recommendation.create()
self.recommendation.populate(recommendations_path)
print("-" * 88)

print(
    f"Creating an EC2 launch template that runs '{startup_script}' when
an instance starts.\n"
    f"This script starts a Python web server defined in the `server.py`
script. The web server\n"
    f"listens to HTTP requests on port 80 and responds to requests to '/'
and to '/healthcheck'.\n"
    f"For demo purposes, this server is run as the root user. In
production, the best practice is to\n"
    f"run a web server, such as Apache, with least-privileged
credentials.\n"
)
print(
    f"The template also defines an IAM policy that each instance uses to
assume a role that grants\n"
    f"permissions to access the DynamoDB recommendation table and Systems
Manager parameters\n"
    f"that control the flow of the demo.\n"
)
self.autoscaler.create_template(startup_script, instance_policy)
print("-" * 88)

print(
    f"Creating an EC2 Auto Scaling group that maintains three EC2
instances, each in a different\n"
    f"Availability Zone."
)
zones = self.autoscaler.create_group(3)
print("-" * 88)
print(
    "At this point, you have EC2 instances created. Once each instance
starts, it listens for\n"
```



```
        "HTTP requests. You can see these instances in the console or
continue with the demo."
    )
    print("-" * 88)
    q.ask("Press Enter when you're ready to continue.")

    print(f"Creating variables that control the flow of the demo.\n")
    self.param_helper.reset()

    print(
        "\nCreating an Elastic Load Balancing target group and load balancer.
The target group\n"
        "defines how the load balancer connects to instances. The load
balancer provides a\n"
        "single endpoint where clients connect and dispatches requests to
instances in the group.\n"
    )
    vpc = self.autoscaler.get_default_vpc()
    subnets = self.autoscaler.get_subnets(vpc["VpcId"], zones)
    target_group = self.loadbalancer.create_target_group(
        self.protocol, self.port, vpc["VpcId"]
    )
    self.loadbalancer.create_load_balancer(
        [subnet["SubnetId"] for subnet in subnets], target_group
    )
    self.autoscaler.attach_load_balancer_target_group(target_group)
    print(f"Verifying access to the load balancer endpoint...")
    lb_success = self.loadbalancer.verify_load_balancer_endpoint()
    if not lb_success:
        print(
            "Couldn't connect to the load balancer, verifying that the port
is open..."
        )
        current_ip_address = requests.get(
            "http://checkip.amazonaws.com"
        ).text.strip()
        sec_group, port_is_open = self.autoscaler.verify_inbound_port(
            vpc, self.port, current_ip_address
        )
        sec_group, ssh_port_is_open = self.autoscaler.verify_inbound_port(
            vpc, self.ssh_port, current_ip_address
        )
        if not port_is_open:
            print(
```

```

        "For this example to work, the default security group for
your default VPC must\n"
        "allows access from this computer. You can either add it
automatically from this\n"
        "example or add it yourself using the AWS Management Console.
\n"
    )
    if q.ask(
        f"Do you want to add a rule to security group
{sec_group['GroupId']} to allow\n"
        f"inbound traffic on port {self.port} from your computer's IP
address of {current_ip_address}? (y/n) ",
        q.is_yesno,
    ):
        self.autoscaler.open_inbound_port(
            sec_group["GroupId"], self.port, current_ip_address
        )
    if not ssh_port_is_open:
        if q.ask(
            f"Do you want to add a rule to security group
{sec_group['GroupId']} to allow\n"
            f"inbound SSH traffic on port {self.ssh_port} for debugging
from your computer's IP address of {current_ip_address}? (y/n) ",
            q.is_yesno,
        ):
            self.autoscaler.open_inbound_port(
                sec_group["GroupId"], self.ssh_port, current_ip_address
            )
    lb_success = self.loadbalancer.verify_load_balancer_endpoint()
    if lb_success:
        print("Your load balancer is ready. You can access it by browsing to:
\n")
        print(f"\thttp://{self.loadbalancer.endpoint()}\n")
    else:
        print(
            "Couldn't get a successful response from the load balancer
endpoint. Troubleshoot by\n"
            "manually verifying that your VPC and security group are
configured correctly and that\n"
            "you can successfully make a GET request to the load balancer
endpoint:\n"
        )
        print(f"\thttp://{self.loadbalancer.endpoint()}\n")
    print("-" * 88)

```

```

q.ask("Press Enter when you're ready to continue with the demo.")

def demo_choices(self):
    actions = [
        "Send a GET request to the load balancer endpoint.",
        "Check the health of load balancer targets.",
        "Go to the next part of the demo.",
    ]
    choice = 0
    while choice != 2:
        print("-" * 88)
        print(
            "\nSee the current state of the service by selecting one of the
following choices:\n"
        )
        choice = q.choose("\nWhich action would you like to take? ", actions)
        print("-" * 88)
        if choice == 0:
            print("Request:\n")
            print(f"GET http://{self.loadbalancer.endpoint()}")
            response = requests.get(f"http://{self.loadbalancer.endpoint()}")
            print("\nResponse:\n")
            print(f"{response.status_code}")
            if response.headers.get("content-type") == "application/json":
                pp(response.json())
        elif choice == 1:
            print("\nChecking the health of load balancer targets:\n")
            health = self.loadbalancer.check_target_health()
            for target in health:
                state = target["TargetHealth"]["State"]
                print(
                    f"\tTarget {target['Target']['Id']} on port
{target['Target']['Port']} is {state}"
                )
                if state != "healthy":
                    print(
                        f"\t\t{target['TargetHealth']['Reason']}:
{target['TargetHealth']['Description']}\n"
                    )
            print(
                f"\nNote that it can take a minute or two for the health
check to update\n"
                f"after changes are made.\n"
            )

```

```
        elif choice == 2:
            print("\nOkay, let's move on.")
            print("-" * 88)

    def demo(self):
        ssm_only_policy = f"{self.resource_path}/ssm_only_policy.json"

        print("\nResetting parameters to starting values for demo.\n")
        self.param_helper.reset()

        print(
            "\nThis part of the demonstration shows how to toggle different parts
of the system\n"
            "to create situations where the web service fails, and shows how
using a resilient\n"
            "architecture can keep the web service running in spite of these
failures."
        )
        print("-" * 88)

        print(
            "At the start, the load balancer endpoint returns recommendations and
reports that all targets are healthy."
        )
        self.demo_choices()

        print(
            f"The web service running on the EC2 instances gets recommendations
by querying a DynamoDB table.\n"
            f"The table name is contained in a Systems Manager parameter named
'{self.param_helper.table}'.\n"
            f"To simulate a failure of the recommendation service, let's set this
parameter to name a non-existent table.\n"
        )
        self.param_helper.put(self.param_helper.table, "this-is-not-a-table")
        print(
            "\nNow, sending a GET request to the load balancer endpoint returns a
failure code. But, the service reports as\n"
            "healthy to the load balancer because shallow health checks don't
check for failure of the recommendation service."
        )
        self.demo_choices()

        print(
```

```
        f"Instead of failing when the recommendation service fails, the web
service can return a static response.\n"
        f"While this is not a perfect solution, it presents the customer with
a somewhat better experience than failure.\n"
    )
    self.param_helper.put(self.param_helper.failure_response, "static")
    print(
        f"\nNow, sending a GET request to the load balancer endpoint returns
a static response.\n"
        f"The service still reports as healthy because health checks are
still shallow.\n"
    )
    self.demo_choices()

    print("Let's reinstate the recommendation service.\n")
    self.param_helper.put(self.param_helper.table,
self.recommendation.table_name)
    print(
        "\nLet's also substitute bad credentials for one of the instances in
the target group so that it can't\n"
        "access the DynamoDB recommendation table.\n"
    )
    self.autoscaler.create_instance_profile(
        ssm_only_policy,
        self.autoscaler.bad_creds_policy_name,
        self.autoscaler.bad_creds_role_name,
        self.autoscaler.bad_creds_profile_name,
        ["AmazonSSMManagedInstanceCore"],
    )
    instances = self.autoscaler.get_instances()
    bad_instance_id = instances[0]
    instance_profile = self.autoscaler.get_instance_profile(bad_instance_id)
    print(
        f"\nReplacing the profile for instance {bad_instance_id} with a
profile that contains\n"
        f"bad credentials...\n"
    )
    self.autoscaler.replace_instance_profile(
        bad_instance_id,
        self.autoscaler.bad_creds_profile_name,
        instance_profile["AssociationId"],
    )
    print(
```

```
        "Now, sending a GET request to the load balancer endpoint returns
either a recommendation or a static response,\n"
        "depending on which instance is selected by the load balancer.\n"
    )
    self.demo_choices()

    print(
        "\nLet's implement a deep health check. For this demo, a deep health
check tests whether\n"
        "the web service can access the DynamoDB table that it depends on for
recommendations. Note that\n"
        "the deep health check is only for ELB routing and not for Auto
Scaling instance health.\n"
        "This kind of deep health check is not recommended for Auto Scaling
instance health, because it\n"
        "risks accidental termination of all instances in the Auto Scaling
group when a dependent service fails.\n"
    )
    print(
        "By implementing deep health checks, the load balancer can detect
when one of the instances is failing\n"
        "and take that instance out of rotation.\n"
    )
    self.param_helper.put(self.param_helper.health_check, "deep")
    print(
        f"\nNow, checking target health indicates that the instance with bad
credentials ({bad_instance_id})\n"
        f"is unhealthy. Note that it might take a minute or two for the load
balancer to detect the unhealthy \n"
        f"instance. Sending a GET request to the load balancer endpoint
always returns a recommendation, because\n"
        "the load balancer takes unhealthy instances out of its rotation.\n"
    )
    self.demo_choices()

    print(
        "\nBecause the instances in this demo are controlled by an auto
scaler, the simplest way to fix an unhealthy\n"
        "instance is to terminate it and let the auto scaler start a new
instance to replace it.\n"
    )
    self.autoscaler.terminate_instance(bad_instance_id)
    print(
```

```
        "\nEven while the instance is terminating and the new instance is
starting, sending a GET\n"
        "request to the web service continues to get a successful
recommendation response because\n"
        "the load balancer routes requests to the healthy instances. After
the replacement instance\n"
        "starts and reports as healthy, it is included in the load balancing
rotation.\n"
        "\nNote that terminating and replacing an instance typically takes
several minutes, during which time you\n"
        "can see the changing health check status until the new instance is
running and healthy.\n"
    )
    self.demo_choices()

    print(
        "\nIf the recommendation service fails now, deep health checks mean
all instances report as unhealthy.\n"
    )
    self.param_helper.put(self.param_helper.table, "this-is-not-a-table")
    print(
        "\nWhen all instances are unhealthy, the load balancer continues to
route requests even to\n"
        "unhealthy instances, allowing them to fail open and return a static
response rather than fail\n"
        "closed and report failure to the customer."
    )
    self.demo_choices()
    self.param_helper.reset()

    def destroy(self):
        print(
            "This concludes the demo of how to build and manage a resilient
service.\n"
            "To keep things tidy and to avoid unwanted charges on your account,
we can clean up all AWS resources\n"
            "that were created for this demo."
        )
        if q.ask("Do you want to clean up all demo resources? (y/n) ",
q.is_yesno):
            self.loadbalancer.delete_load_balancer()
            self.loadbalancer.delete_target_group()
            self.autoscaler.delete_group()
            self.autoscaler.delete_key_pair()
```

```
        self.autoscaler.delete_template()
        self.autoscaler.delete_instance_profile(
            self.autoscaler.bad_creds_profile_name,
            self.autoscaler.bad_creds_role_name,
        )
        self.recommendation.destroy()
    else:
        print(
            "Okay, we'll leave the resources intact.\n"
            "Don't forget to delete them when you're done with them or you
            might incur unexpected charges."
        )

def main():
    parser = argparse.ArgumentParser()
    parser.add_argument(
        "--action",
        required=True,
        choices=["all", "deploy", "demo", "destroy"],
        help="The action to take for the demo. When 'all' is specified, resources
        are\n"
        "deployed, the demo is run, and resources are destroyed.",
    )
    parser.add_argument(
        "--resource_path",
        default="../../../workflows/resilient_service/resources",
        help="The path to resource files used by this example, such as IAM
        policies and\n"
        "instance scripts.",
    )
    args = parser.parse_args()

    print("-" * 88)
    print(
        "Welcome to the demonstration of How to Build and Manage a Resilient
        Service!"
    )
    print("-" * 88)

    prefix = "doc-example-resilience"
    recommendation = RecommendationService.from_client(
        "doc-example-recommendation-service"
    )
```



```

autoscaler = AutoScaler.from_client(prefix)
loadbalancer = LoadBalancer.from_client(prefix)
param_helper = ParameterHelper.from_client(recommendation.table_name)
runner = Runner(
    args.resource_path, recommendation, autoscaler, loadbalancer,
param_helper
)
actions = [args.action] if args.action != "all" else ["deploy", "demo",
"destroy"]
for action in actions:
    if action == "deploy":
        runner.deploy()
    elif action == "demo":
        runner.demo()
    elif action == "destroy":
        runner.destroy()

print("-" * 88)
print("Thanks for watching!")
print("-" * 88)

if __name__ == "__main__":
    logging.basicConfig(level=logging.INFO, format="%(levelname)s: %(message)s")
    main()

```

Cree una clase que agrupe las acciones de escalado automático y Amazon EC2.

```

class AutoScaler:
    """
    Encapsulates Amazon EC2 Auto Scaling and EC2 management actions.
    """

    def __init__(
        self,
        resource_prefix,
        inst_type,
        ami_param,
        autoscaling_client,
        ec2_client,
        ssm_client,
        iam_client,

```

```

    ):
        """
        :param resource_prefix: The prefix for naming AWS resources that are
        created by this class.
        :param inst_type: The type of EC2 instance to create, such as t3.micro.
        :param ami_param: The Systems Manager parameter used to look up the AMI
        that is
                created.
        :param autoscaling_client: A Boto3 EC2 Auto Scaling client.
        :param ec2_client: A Boto3 EC2 client.
        :param ssm_client: A Boto3 Systems Manager client.
        :param iam_client: A Boto3 IAM client.
        """
        self.inst_type = inst_type
        self.ami_param = ami_param
        self.autoscaling_client = autoscaling_client
        self.ec2_client = ec2_client
        self.ssm_client = ssm_client
        self.iam_client = iam_client
        self.launch_template_name = f"{resource_prefix}-template"
        self.group_name = f"{resource_prefix}-group"
        self.instance_policy_name = f"{resource_prefix}-pol"
        self.instance_role_name = f"{resource_prefix}-role"
        self.instance_profile_name = f"{resource_prefix}-prof"
        self.bad_creds_policy_name = f"{resource_prefix}-bc-pol"
        self.bad_creds_role_name = f"{resource_prefix}-bc-role"
        self.bad_creds_profile_name = f"{resource_prefix}-bc-prof"
        self.key_pair_name = f"{resource_prefix}-key-pair"

    @classmethod
    def from_client(cls, resource_prefix):
        """
        Creates this class from Boto3 clients.

        :param resource_prefix: The prefix for naming AWS resources that are
        created by this class.
        """
        as_client = boto3.client("autoscaling")
        ec2_client = boto3.client("ec2")
        ssm_client = boto3.client("ssm")
        iam_client = boto3.client("iam")
        return cls(
            resource_prefix,

```

```

        "t3.micro",
        "/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2",
        as_client,
        ec2_client,
        ssm_client,
        iam_client,
    )

    def create_instance_profile(
        self, policy_file, policy_name, role_name, profile_name,
        aws_managed_policies=()
    ):
        """
        Creates a policy, role, and profile that is associated with instances
        created by
        this class. An instance's associated profile defines a role that is
        assumed by the
        instance. The role has attached policies that specify the AWS permissions
        granted to
        clients that run on the instance.

        :param policy_file: The name of a JSON file that contains the policy
        definition to
            create and attach to the role.
        :param policy_name: The name to give the created policy.
        :param role_name: The name to give the created role.
        :param profile_name: The name to the created profile.
        :param aws_managed_policies: Additional AWS-managed policies that are
        attached to
            the role, such as
        AmazonSSMManagedInstanceCore to grant
            use of Systems Manager to send commands to
        the instance.

        :return: The ARN of the profile that is created.
        """
        assume_role_doc = {
            "Version": "2012-10-17",
            "Statement": [
                {
                    "Effect": "Allow",
                    "Principal": {"Service": "ec2.amazonaws.com"},
                    "Action": "sts:AssumeRole",
                }
            ],
        ],

```

```

    }
    with open(policy_file) as file:
        instance_policy_doc = file.read()

    policy_arn = None
    try:
        pol_response = self.iam_client.create_policy(
            PolicyName=policy_name, PolicyDocument=instance_policy_doc
        )
        policy_arn = pol_response["Policy"]["Arn"]
        log.info("Created policy with ARN %s.", policy_arn)
    except ClientError as err:
        if err.response["Error"]["Code"] == "EntityAlreadyExists":
            log.info("Policy %s already exists, nothing to do.", policy_name)
            list_pol_response = self.iam_client.list_policies(Scope="Local")
            for pol in list_pol_response["Policies"]:
                if pol["PolicyName"] == policy_name:
                    policy_arn = pol["Arn"]
                    break
        if policy_arn is None:
            raise AutoScalerError(f"Couldn't create policy {policy_name}:
{err}")

    try:
        self.iam_client.create_role(
            RoleName=role_name,
            AssumeRolePolicyDocument=json.dumps(assume_role_doc)
        )
        self.iam_client.attach_role_policy(RoleName=role_name,
            PolicyArn=policy_arn)
        for aws_policy in aws_managed_policies:
            self.iam_client.attach_role_policy(
                RoleName=role_name,
                PolicyArn=f"arn:aws:iam::aws:policy/{aws_policy}",
            )
        log.info("Created role %s and attached policy %s.", role_name,
            policy_arn)
    except ClientError as err:
        if err.response["Error"]["Code"] == "EntityAlreadyExists":
            log.info("Role %s already exists, nothing to do.", role_name)
        else:
            raise AutoScalerError(f"Couldn't create role {role_name}: {err}")

    try:

```

```

        profile_response = self.iam_client.create_instance_profile(
            InstanceProfileName=profile_name
        )
        waiter = self.iam_client.get_waiter("instance_profile_exists")
        waiter.wait(InstanceProfileName=profile_name)
        time.sleep(10) # wait a little longer
        profile_arn = profile_response["InstanceProfile"]["Arn"]
        self.iam_client.add_role_to_instance_profile(
            InstanceProfileName=profile_name, RoleName=role_name
        )
        log.info("Created profile %s and added role %s.", profile_name,
role_name)
    except ClientError as err:
        if err.response["Error"]["Code"] == "EntityAlreadyExists":
            prof_response = self.iam_client.get_instance_profile(
                InstanceProfileName=profile_name
            )
            profile_arn = prof_response["InstanceProfile"]["Arn"]
            log.info(
                "Instance profile %s already exists, nothing to do.",
profile_name
            )
        else:
            raise AutoScalerError(
                f"Couldn't create profile {profile_name} and attach it to
role\n"
                f"{role_name}: {err}"
            )
    return profile_arn

def get_instance_profile(self, instance_id):
    """
    Gets data about the profile associated with an instance.

    :param instance_id: The ID of the instance to look up.
    :return: The profile data.
    """
    try:
        response =
self.ec2_client.describe_iam_instance_profile_associations(
            Filters=[{"Name": "instance-id", "Values": [instance_id]}]
        )
    except ClientError as err:

```

```

        raise AutoScalerError(
            f"Couldn't get instance profile association for instance
{instance_id}: {err}"
        )
    else:
        return response["IamInstanceProfileAssociations"][0]

def replace_instance_profile(
    self, instance_id, new_instance_profile_name, profile_association_id
):
    """
    Replaces the profile associated with a running instance. After the
profile is
replaced, the instance is rebooted to ensure that it uses the new
profile. When
the instance is ready, Systems Manager is used to restart the Python web
server.

:param instance_id: The ID of the instance to update.
:param new_instance_profile_name: The name of the new profile to
associate with
                               the specified instance.
:param profile_association_id: The ID of the existing profile association
for the
                               instance.
    """
    try:
        self.ec2_client.replace_iam_instance_profile_association(
            IamInstanceProfile={"Name": new_instance_profile_name},
            AssociationId=profile_association_id,
        )
        log.info(
            "Replaced instance profile for association %s with profile %s.",
            profile_association_id,
            new_instance_profile_name,
        )
        time.sleep(5)
        inst_ready = False
        tries = 0
        while not inst_ready:
            if tries % 6 == 0:
                self.ec2_client.reboot_instances(InstanceIds=[instance_id])
                log.info(

```

```

        "Rebooting instance %s and waiting for it to to be
ready.",
        instance_id,
    )
    tries += 1
    time.sleep(10)
    response = self.ssm_client.describe_instance_information()
    for info in response["InstanceInformationList"]:
        if info["InstanceId"] == instance_id:
            inst_ready = True
    self.ssm_client.send_command(
        InstanceIds=[instance_id],
        DocumentName="AWS-RunShellScript",
        Parameters={"commands": ["cd / && sudo python3 server.py 80"]},
    )
    log.info("Restarted the Python web server on instance %s.",
instance_id)
    except ClientError as err:
        raise AutoScalerError(
            f"Couldn't replace instance profile for association
{profile_association_id}: {err}"
        )

def delete_instance_profile(self, profile_name, role_name):
    """
    Detaches a role from an instance profile, detaches policies from the
role,
and deletes all the resources.

:param profile_name: The name of the profile to delete.
:param role_name: The name of the role to delete.
    """
    try:
        self.iam_client.remove_role_from_instance_profile(
            InstanceProfileName=profile_name, RoleName=role_name
        )

self.iam_client.delete_instance_profile(InstanceProfileName=profile_name)
    log.info("Deleted instance profile %s.", profile_name)
    attached_policies = self.iam_client.list_attached_role_policies(
        RoleName=role_name
    )
    for pol in attached_policies["AttachedPolicies"]:
```

```

        self.iam_client.detach_role_policy(
            RoleName=role_name, PolicyArn=pol["PolicyArn"]
        )
        if not pol["PolicyArn"].startswith("arn:aws:iam::aws"):
            self.iam_client.delete_policy(PolicyArn=pol["PolicyArn"])
            log.info("Detached and deleted policy %s.", pol["PolicyName"])
        self.iam_client.delete_role(RoleName=role_name)
        log.info("Deleted role %s.", role_name)
    except ClientError as err:
        if err.response["Error"]["Code"] == "NoSuchEntity":
            log.info(
                "Instance profile %s doesn't exist, nothing to do.",
profile_name
            )
        else:
            raise AutoScalerError(
                f"Couldn't delete instance profile {profile_name} or detach "
                f"policies and delete role {role_name}: {err}"
            )

def create_key_pair(self, key_pair_name):
    """
    Creates a new key pair.

    :param key_pair_name: The name of the key pair to create.
    :return: The newly created key pair.
    """
    try:
        response = self.ec2_client.create_key_pair(KeyName=key_pair_name)
        with open(f"{key_pair_name}.pem", "w") as file:
            file.write(response["KeyMaterial"])
        chmod(f"{key_pair_name}.pem", 0o600)
        log.info("Created key pair %s.", key_pair_name)
    except ClientError as err:
        raise AutoScalerError(f"Couldn't create key pair {key_pair_name}:
{err}")

def delete_key_pair(self):
    """
    Deletes a key pair.

    :param key_pair_name: The name of the key pair to delete.

```



```

    """
    try:
        self.ec2_client.delete_key_pair(KeyName=self.key_pair_name)
        remove(f"{self.key_pair_name}.pem")
        log.info("Deleted key pair %s.", self.key_pair_name)
    except ClientError as err:
        raise AutoScalerError(
            f"Couldn't delete key pair {self.key_pair_name}: {err}"
        )
    except FileNotFoundError:
        log.info("Key pair %s doesn't exist, nothing to do.",
self.key_pair_name)
    except PermissionError:
        log.info(
            "Inadequate permissions to delete key pair %s.",
self.key_pair_name
        )
    except Exception as err:
        raise AutoScalerError(
            f"Couldn't delete key pair {self.key_pair_name}: {err}"
        )

    def create_template(self, server_startup_script_file, instance_policy_file):
        """
        Creates an Amazon EC2 launch template to use with Amazon EC2 Auto
        Scaling. The
        launch template specifies a Bash script in its user data field that runs
        after
        the instance is started. This script installs Python packages and starts
        a
        Python web server on the instance.

        :param server_startup_script_file: The path to a Bash script file that is
run
                                         when an instance starts.
        :param instance_policy_file: The path to a file that defines a
permissions policy
                                         to create and attach to the instance
profile.
        :return: Information about the newly created template.
        """
        template = {}
        try:

```

```
self.create_key_pair(self.key_pair_name)
self.create_instance_profile(
    instance_policy_file,
    self.instance_policy_name,
    self.instance_role_name,
    self.instance_profile_name,
)
with open(server_startup_script_file) as file:
    start_server_script = file.read()
ami_latest = self.ssm_client.get_parameter(Name=self.ami_param)
ami_id = ami_latest["Parameter"]["Value"]
lt_response = self.ec2_client.create_launch_template(
    LaunchTemplateName=self.launch_template_name,
    LaunchTemplateData={
        "InstanceType": self.inst_type,
        "ImageId": ami_id,
        "IamInstanceProfile": {"Name": self.instance_profile_name},
        "UserData": base64.b64encode(
            start_server_script.encode(encoding="utf-8")
        ).decode(encoding="utf-8"),
        "KeyName": self.key_pair_name,
    },
)
template = lt_response["LaunchTemplate"]
log.info(
    "Created launch template %s for AMI %s on %s.",
    self.launch_template_name,
    ami_id,
    self.inst_type,
)
except ClientError as err:
    if (
        err.response["Error"]["Code"]
        == "InvalidLaunchTemplateName.AlreadyExistsException"
    ):
        log.info(
            "Launch template %s already exists, nothing to do.",
            self.launch_template_name,
        )
    else:
        raise AutoScalerError(
            f"Couldn't create launch template
{self.launch_template_name}: {err}."
        )
```

```
return template

def delete_template(self):
    """
    Deletes a launch template.
    """
    try:
        self.ec2_client.delete_launch_template(
            LaunchTemplateName=self.launch_template_name
        )
        self.delete_instance_profile(
            self.instance_profile_name, self.instance_role_name
        )
        log.info("Launch template %s deleted.", self.launch_template_name)
    except ClientError as err:
        if (
            err.response["Error"]["Code"]
            == "InvalidLaunchTemplateName.NotFoundException"
        ):
            log.info(
                "Launch template %s does not exist, nothing to do.",
                self.launch_template_name,
            )
        else:
            raise AutoScalerError(
                f"Couldn't delete launch template
{self.launch_template_name}: {err}."
            )

def get_availability_zones(self):
    """
    Gets a list of Availability Zones in the AWS Region of the Amazon EC2
    client.

    :return: The list of Availability Zones for the client Region.
    """
    try:
        response = self.ec2_client.describe_availability_zones()
        zones = [zone["ZoneName"] for zone in response["AvailabilityZones"]]
    except ClientError as err:
        raise AutoScalerError(f"Couldn't get availability zones: {err}.")
    else:
```

```
        return zones

def create_group(self, group_size):
    """
    Creates an EC2 Auto Scaling group with the specified size.

    :param group_size: The number of instances to set for the minimum and
maximum in
                        the group.
    :return: The list of Availability Zones specified for the group.
    """
    zones = []
    try:
        zones = self.get_availability_zones()
        self.autoscaling_client.create_auto_scaling_group(
            AutoScalingGroupName=self.group_name,
            AvailabilityZones=zones,
            LaunchTemplate={
                "LaunchTemplateName": self.launch_template_name,
                "Version": "$Default",
            },
            MinSize=group_size,
            MaxSize=group_size,
        )
        log.info(
            "Created EC2 Auto Scaling group %s with availability zones %s.",
            self.launch_template_name,
            zones,
        )
    except ClientError as err:
        if err.response["Error"]["Code"] == "AlreadyExists":
            log.info(
                "EC2 Auto Scaling group %s already exists, nothing to do.",
                self.group_name,
            )
        else:
            raise AutoScalerError(
                f"Couldn't create EC2 Auto Scaling group {self.group_name}:
{err}")
    return zones
```

```
def get_instances(self):
    """
    Gets data about the instances in the EC2 Auto Scaling group.

    :return: Data about the instances.
    """
    try:
        as_response = self.autoscaling_client.describe_auto_scaling_groups(
            AutoScalingGroupNames=[self.group_name]
        )
        instance_ids = [
            i["InstanceId"]
            for i in as_response["AutoScalingGroups"][0]["Instances"]
        ]
    except ClientError as err:
        raise AutoScalerError(
            f"Couldn't get instances for Auto Scaling group
{self.group_name}: {err}")
    else:
        return instance_ids

def terminate_instance(self, instance_id):
    """
    Terminates and instances in an EC2 Auto Scaling group. After an instance
is
    terminated, it can no longer be accessed.

    :param instance_id: The ID of the instance to terminate.
    """
    try:
        self.autoscaling_client.terminate_instance_in_auto_scaling_group(
            InstanceId=instance_id, ShouldDecrementDesiredCapacity=False
        )
        log.info("Terminated instance %s.", instance_id)
    except ClientError as err:
        raise AutoScalerError(f"Couldn't terminate instance {instance_id}:
{err}")

def attach_load_balancer_target_group(self, lb_target_group):
    """
    Attaches an Elastic Load Balancing (ELB) target group to this EC2 Auto
Scaling group.
```

```
    The target group specifies how the load balancer forward requests to the
instances
    in the group.

:param lb_target_group: Data about the ELB target group to attach.
"""
try:
    self.autoscaling_client.attach_load_balancer_target_groups(
        AutoScalingGroupName=self.group_name,
        TargetGroupARNs=[lb_target_group["TargetGroupArn"]],
    )
    log.info(
        "Attached load balancer target group %s to auto scaling group
%s.",
        lb_target_group["TargetGroupName"],
        self.group_name,
    )
except ClientError as err:
    raise AutoScalerError(
        f"Couldn't attach load balancer target group
{lb_target_group['TargetGroupName']}\n"
        f"to auto scaling group {self.group_name}"
    )

def _try_terminate_instance(self, inst_id):
    stopping = False
    log.info(f"Stopping {inst_id}.")
    while not stopping:
        try:
            self.autoscaling_client.terminate_instance_in_auto_scaling_group(
                InstanceId=inst_id, ShouldDecrementDesiredCapacity=True
            )
            stopping = True
        except ClientError as err:
            if err.response["Error"]["Code"] == "ScalingActivityInProgress":
                log.info("Scaling activity in progress for %s. Waiting...",
inst_id)
                time.sleep(10)
            else:
                raise AutoScalerError(f"Couldn't stop instance {inst_id}:
{err}.")

def _try_delete_group(self):
```

```
    """
    Tries to delete the EC2 Auto Scaling group. If the group is in use or in
    progress,
    the function waits and retries until the group is successfully deleted.
    """
    stopped = False
    while not stopped:
        try:
            self.autoscaling_client.delete_auto_scaling_group(
                AutoScalingGroupName=self.group_name
            )
            stopped = True
            log.info("Deleted EC2 Auto Scaling group %s.", self.group_name)
        except ClientError as err:
            if (
                err.response["Error"]["Code"] == "ResourceInUse"
                or err.response["Error"]["Code"] ==
                "ScalingActivityInProgress"
            ):
                log.info(
                    "Some instances are still running. Waiting for them to
                    stop..."
                )
                time.sleep(10)
            else:
                raise AutoScalerError(
                    f"Couldn't delete group {self.group_name}: {err}."
                )

    def delete_group(self):
        """
        Terminates all instances in the group, deletes the EC2 Auto Scaling
        group.
        """
        try:
            response = self.autoscaling_client.describe_auto_scaling_groups(
                AutoScalingGroupNames=[self.group_name]
            )
            groups = response.get("AutoScalingGroups", [])
            if len(groups) > 0:
                self.autoscaling_client.update_auto_scaling_group(
                    AutoScalingGroupName=self.group_name, MinSize=0
                )
```

```
        instance_ids = [inst["InstanceId"] for inst in groups[0]
["Instances"]]
        for inst_id in instance_ids:
            self._try_terminate_instance(inst_id)
            self._try_delete_group()
        else:
            log.info("No groups found named %s, nothing to do.",
self.group_name)
        except ClientError as err:
            raise AutoScalerError(f"Couldn't delete group {self.group_name}:
{err}.")

def get_default_vpc(self):
    """
    Gets the default VPC for the account.

    :return: Data about the default VPC.
    """
    try:
        response = self.ec2_client.describe_vpcs(
            Filters=[{"Name": "is-default", "Values": ["true"]}])
    except ClientError as err:
        raise AutoScalerError(f"Couldn't get default VPC: {err}")
    else:
        return response["Vpcs"][0]

def verify_inbound_port(self, vpc, port, ip_address):
    """
    Verify the default security group of the specified VPC allows ingress
from this
    computer. This can be done by allowing ingress from this computer's IP
address. In some situations, such as connecting from a corporate network,
you
    must instead specify a prefix list ID. You can also temporarily open the
port to
    any IP address while running this example. If you do, be sure to remove
public
    access when you're done.

    :param vpc: The VPC used by this example.
    :param port: The port to verify.
```



```

:param ip_address: This computer's IP address.
:return: The default security group of the specific VPC, and a value that
indicates
        whether the specified port is open.
"""
try:
    response = self.ec2_client.describe_security_groups(
        Filters=[
            {"Name": "group-name", "Values": ["default"]},
            {"Name": "vpc-id", "Values": [vpc["VpcId"]]},
        ]
    )
    sec_group = response["SecurityGroups"][0]
    port_is_open = False
    log.info("Found default security group %s.", sec_group["GroupId"])
    for ip_perm in sec_group["IpPermissions"]:
        if ip_perm.get("FromPort", 0) == port:
            log.info("Found inbound rule: %s", ip_perm)
            for ip_range in ip_perm["IpRanges"]:
                cidr = ip_range.get("CidrIp", "")
                if cidr.startswith(ip_address) or cidr == "0.0.0.0/0":
                    port_is_open = True
            if ip_perm["PrefixListIds"]:
                port_is_open = True
            if not port_is_open:
                log.info(
                    "The inbound rule does not appear to be open to
either this computer's IP\n"
                    "address of %s, to all IP addresses (0.0.0.0/0), or
to a prefix list ID.",
                    ip_address,
                )
            else:
                break
    except ClientError as err:
        raise AutoScalerError(
            f"Couldn't verify inbound rule for port {port} for VPC
{vpc['VpcId']}: {err}"
        )
    else:
        return sec_group, port_is_open

def open_inbound_port(self, sec_group_id, port, ip_address):

```

```
"""
Add an ingress rule to the specified security group that allows access on
the
specified port from the specified IP address.

:param sec_group_id: The ID of the security group to modify.
:param port: The port to open.
:param ip_address: The IP address that is granted access.
"""
try:
    self.ec2_client.authorize_security_group_ingress(
        GroupId=sec_group_id,
        CidrIp=f"{ip_address}/32",
        FromPort=port,
        ToPort=port,
        IpProtocol="tcp",
    )
    log.info(
        "Authorized ingress to %s on port %s from %s.",
        sec_group_id,
        port,
        ip_address,
    )
except ClientError as err:
    raise AutoScalerError(
        f"Couldn't authorize ingress to {sec_group_id} on port {port}
from {ip_address}: {err}"
    )

def get_subnets(self, vpc_id, zones):
    """
    Gets the default subnets in a VPC for a specified list of Availability
    Zones.

    :param vpc_id: The ID of the VPC to look up.
    :param zones: The list of Availability Zones to look up.
    :return: The list of subnets found.
    """
    try:
        response = self.ec2_client.describe_subnets(
            Filters=[
                {"Name": "vpc-id", "Values": [vpc_id]},
                {"Name": "availability-zone", "Values": zones},
            ]
        )
```

```

        {"Name": "default-for-az", "Values": ["true"]},
    ]
)
subnets = response["Subnets"]
log.info("Found %s subnets for the specified zones.", len(subnets))
except ClientError as err:
    raise AutoScalerError(f"Couldn't get subnets: {err}")
else:
    return subnets

```

Cree una clase que resuma las acciones de Elastic Load Balancing.

```

class LoadBalancer:
    """Encapsulates Elastic Load Balancing (ELB) actions."""

    def __init__(self, target_group_name, load_balancer_name, elb_client):
        """
        :param target_group_name: The name of the target group associated with
        the load balancer.
        :param load_balancer_name: The name of the load balancer.
        :param elb_client: A Boto3 Elastic Load Balancing client.
        """
        self.target_group_name = target_group_name
        self.load_balancer_name = load_balancer_name
        self.elb_client = elb_client
        self._endpoint = None

    @classmethod
    def from_client(cls, resource_prefix):
        """
        Creates this class from a Boto3 client.

        :param resource_prefix: The prefix to give to AWS resources created by
        this class.
        """
        elb_client = boto3.client("elbv2")
        return cls(f"{resource_prefix}-tg", f"{resource_prefix}-lb", elb_client)

```

```
def endpoint(self):
    """
    Gets the HTTP endpoint of the load balancer.

    :return: The endpoint.
    """
    if self._endpoint is None:
        try:
            response = self.elb_client.describe_load_balancers(
                Names=[self.load_balancer_name]
            )
            self._endpoint = response["LoadBalancers"][0]["DNSName"]
        except ClientError as err:
            raise LoadBalancerError(
                f"Couldn't get the endpoint for load balancer
                {self.load_balancer_name}: {err}")
        return self._endpoint

def create_target_group(self, protocol, port, vpc_id):
    """
    Creates an Elastic Load Balancing target group. The target group
    specifies how
    the load balancer forward requests to instances in the group and how
    instance
    health is checked.

    To speed up this demo, the health check is configured with shortened
    times and
    lower thresholds. In production, you might want to decrease the
    sensitivity of
    your health checks to avoid unwanted failures.

    :param protocol: The protocol to use to forward requests, such as 'HTTP'.
    :param port: The port to use to forward requests, such as 80.
    :param vpc_id: The ID of the VPC in which the load balancer exists.
    :return: Data about the newly created target group.
    """
    try:
        response = self.elb_client.create_target_group(
            Name=self.target_group_name,
            Protocol=protocol,
            Port=port,
```

```

        HealthCheckPath="/healthcheck",
        HealthCheckIntervalSeconds=10,
        HealthCheckTimeoutSeconds=5,
        HealthyThresholdCount=2,
        UnhealthyThresholdCount=2,
        VpcId=vpc_id,
    )
    target_group = response["TargetGroups"][0]
    log.info("Created load balancing target group %s.",
self.target_group_name)
    except ClientError as err:
        raise LoadBalancerError(
            f"Couldn't create load balancing target group
{self.target_group_name}: {err}"
        )
    else:
        return target_group

def delete_target_group(self):
    """
    Deletes the target group.
    """
    done = False
    while not done:
        try:
            response = self.elb_client.describe_target_groups(
                Names=[self.target_group_name]
            )
            tg_arn = response["TargetGroups"][0]["TargetGroupArn"]
            self.elb_client.delete_target_group(TargetGroupArn=tg_arn)
            log.info(
                "Deleted load balancing target group %s.",
self.target_group_name
            )
            done = True
        except ClientError as err:
            if err.response["Error"]["Code"] == "TargetGroupNotFound":
                log.info(
                    "Load balancer target group %s not found, nothing to
do.",
                    self.target_group_name,
                )
            done = True

```

```

        elif err.response["Error"]["Code"] == "ResourceInUse":
            log.info(
                "Target group not yet released from load balancer,
waiting..."
            )
            time.sleep(10)
        else:
            raise LoadBalancerError(
                f"Couldn't delete load balancing target group
{self.target_group_name}: {err}"
            )

    def create_load_balancer(self, subnet_ids, target_group):
        """
        Creates an Elastic Load Balancing load balancer that uses the specified
subnets
and forwards requests to the specified target group.

:param subnet_ids: A list of subnets to associate with the load balancer.
:param target_group: An existing target group that is added as a listener
to the
                load balancer.
:return: Data about the newly created load balancer.
        """
        try:
            response = self.elb_client.create_load_balancer(
                Name=self.load_balancer_name, Subnets=subnet_ids
            )
            load_balancer = response["LoadBalancers"][0]
            log.info("Created load balancer %s.", self.load_balancer_name)
            waiter = self.elb_client.get_waiter("load_balancer_available")
            log.info("Waiting for load balancer to be available...")
            waiter.wait(Names=[self.load_balancer_name])
            log.info("Load balancer is available!")
            self.elb_client.create_listener(
                LoadBalancerArn=load_balancer["LoadBalancerArn"],
                Protocol=target_group["Protocol"],
                Port=target_group["Port"],
                DefaultActions=[
                    {
                        "Type": "forward",
                        "TargetGroupArn": target_group["TargetGroupArn"],
                    }
                ]
            )

```

```

        ],
    )
    log.info(
        "Created listener to forward traffic from load balancer %s to
target group %s.",
        self.load_balancer_name,
        target_group["TargetGroupName"],
    )
except ClientError as err:
    raise LoadBalancerError(
        f"Failed to create load balancer {self.load_balancer_name}"
        f"and add a listener for target group
{target_group['TargetGroupName']}: {err}"
    )
else:
    self._endpoint = load_balancer["DNSName"]
    return load_balancer

def delete_load_balancer(self):
    """
    Deletes a load balancer.
    """
    try:
        response = self.elb_client.describe_load_balancers(
            Names=[self.load_balancer_name]
        )
        lb_arn = response["LoadBalancers"][0]["LoadBalancerArn"]
        self.elb_client.delete_load_balancer(LoadBalancerArn=lb_arn)
        log.info("Deleted load balancer %s.", self.load_balancer_name)
        waiter = self.elb_client.get_waiter("load_balancers_deleted")
        log.info("Waiting for load balancer to be deleted...")
        waiter.wait(Names=[self.load_balancer_name])
    except ClientError as err:
        if err.response["Error"]["Code"] == "LoadBalancerNotFound":
            log.info(
                "Load balancer %s does not exist, nothing to do.",
                self.load_balancer_name,
            )
        else:
            raise LoadBalancerError(
                f"Couldn't delete load balancer {self.load_balancer_name}:
{err}"
            )

```

```
def verify_load_balancer_endpoint(self):
    """
    Verify this computer can successfully send a GET request to the load
    balancer endpoint.
    """
    success = False
    retries = 3
    while not success and retries > 0:
        try:
            lb_response = requests.get(f"http://{self.endpoint()}")
            log.info(
                "Got response %s from load balancer endpoint.",
                lb_response.status_code,
            )
            if lb_response.status_code == 200:
                success = True
            else:
                retries = 0
        except requests.exceptions.ConnectionError:
            log.info(
                "Got connection error from load balancer endpoint,
retrying..."
            )
            retries -= 1
            time.sleep(10)
    return success

def check_target_health(self):
    """
    Checks the health of the instances in the target group.

    :return: The health status of the target group.
    """
    try:
        tg_response = self.elb_client.describe_target_groups(
            Names=[self.target_group_name]
        )
        health_response = self.elb_client.describe_target_health(
            TargetGroupArn=tg_response["TargetGroups"][0]["TargetGroupArn"]
        )
    except ClientError as err:
        raise LoadBalancerError(
```



```

        f"Couldn't check health of {self.target_group_name} targets:
{err}"
    )
    else:
        return health_response["TargetHealthDescriptions"]

```

Cree una clase que utilice DynamoDB para simular un servicio de recomendaciones.

```

class RecommendationService:
    """
    Encapsulates a DynamoDB table to use as a service that recommends books,
    movies,
    and songs.
    """

    def __init__(self, table_name, dynamodb_client):
        """
        :param table_name: The name of the DynamoDB recommendations table.
        :param dynamodb_client: A Boto3 DynamoDB client.
        """
        self.table_name = table_name
        self.dynamodb_client = dynamodb_client

    @classmethod
    def from_client(cls, table_name):
        """
        Creates this class from a Boto3 client.

        :param table_name: The name of the DynamoDB recommendations table.
        """
        ddb_client = boto3.client("dynamodb")
        return cls(table_name, ddb_client)

    def create(self):
        """
        Creates a DynamoDB table to use a recommendation service. The table has a
        hash key named 'MediaType' that defines the type of media recommended,
        such as

```

```

        Book or Movie, and a range key named 'ItemId' that, combined with the
        MediaType,
        forms a unique identifier for the recommended item.

        :return: Data about the newly created table.
        """
    try:
        response = self.dynamodb_client.create_table(
            TableName=self.table_name,
            AttributeDefinitions=[
                {"AttributeName": "MediaType", "AttributeType": "S"},
                {"AttributeName": "ItemId", "AttributeType": "N"},
            ],
            KeySchema=[
                {"AttributeName": "MediaType", "KeyType": "HASH"},
                {"AttributeName": "ItemId", "KeyType": "RANGE"},
            ],
            ProvisionedThroughput={"ReadCapacityUnits": 5,
"WriteCapacityUnits": 5},
        )
        log.info("Creating table %s...", self.table_name)
        waiter = self.dynamodb_client.get_waiter("table_exists")
        waiter.wait(TableName=self.table_name)
        log.info("Table %s created.", self.table_name)
    except ClientError as err:
        if err.response["Error"]["Code"] == "ResourceInUseException":
            log.info("Table %s exists, nothing to be do.", self.table_name)
        else:
            raise RecommendationServiceError(
                self.table_name, f"ClientError when creating table: {err}."
            )
    else:
        return response

def populate(self, data_file):
    """
    Populates the recommendations table from a JSON file.

    :param data_file: The path to the data file.
    """
    try:
        with open(data_file) as data:
            items = json.load(data)
            batch = [{"PutRequest": {"Item": item}} for item in items]

```

```

        self.dynamodb_client.batch_write_item(RequestItems={self.table_name:
batch})
        log.info(
            "Populated table %s with items from %s.", self.table_name,
data_file
        )
    except ClientError as err:
        raise RecommendationServiceError(
            self.table_name, f"Couldn't populate table from {data_file}:
{err}"
        )

    def destroy(self):
        """
        Deletes the recommendations table.
        """
        try:
            self.dynamodb_client.delete_table(TableName=self.table_name)
            log.info("Deleting table %s...", self.table_name)
            waiter = self.dynamodb_client.get_waiter("table_not_exists")
            waiter.wait(TableName=self.table_name)
            log.info("Table %s deleted.", self.table_name)
        except ClientError as err:
            if err.response["Error"]["Code"] == "ResourceNotFoundException":
                log.info("Table %s does not exist, nothing to do.",
self.table_name)
            else:
                raise RecommendationServiceError(
                    self.table_name, f"ClientError when deleting table: {err}."
                )

```

Cree una clase que agrupe las acciones de 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 = "doc-example-resilient-architecture-table"
failure_response = "doc-example-resilient-architecture-failure-response"
health_check = "doc-example-resilient-architecture-health-check"

def __init__(self, table_name, ssm_client):
    """
    :param table_name: The name of the DynamoDB table that is used as a
    recommendation
                        service.
    :param ssm_client: A Boto3 Systems Manager client.
    """
    self.ssm_client = ssm_client
    self.table_name = table_name

    @classmethod
    def from_client(cls, table_name):
        ssm_client = boto3.client("ssm")
        return cls(table_name, ssm_client)

    def reset(self):
        """
        Resets the Systems Manager parameters to starting values for the demo.
        These are the name of the DynamoDB recommendation table, no response when
a
        dependency fails, and shallow health checks.
        """
        self.put(self.table, self.table_name)
        self.put(self.failure_response, "none")
        self.put(self.health_check, "shallow")

    def put(self, name, value):
        """
        Sets the value of a named Systems Manager parameter.

        :param name: The name of the parameter.
        :param value: The new value of the parameter.
        """
        try:
            self.ssm_client.put_parameter(
                Name=name, Value=value, Overwrite=True, Type="String"
            )
            log.info("Setting demo parameter %s to '%s'.", name, value)

```

```
except ClientError as err:
    raise ParameterHelperError(
        f"Couldn't set parameter {name} to {value}: {err}"
    )
```

- Para obtener información sobre la API, consulte los siguientes temas en la Referencia de la API de AWS SDK para Python (Boto3).
  - [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 obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte [Cree recursos de Amazon EC2 mediante un SDK AWS](#). En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

## Comience a utilizar las instancias de Amazon EC2 mediante un SDK AWS

En el siguiente ejemplo de código, se muestra cómo:

- Cree un par de claves y un grupo de seguridad.
- Seleccione una Imagen de máquina de Amazon (AMI) y un tipo de instancia; a continuación, cree una instancia.
- Detenga y vuelva a iniciar la instancia.
- Asocie una dirección IP elástica a su instancia.
- Conéctese a tu instancia con SSH y, a continuación, limpie los recursos.

.NET

AWS SDK for .NET

### Note

Hay más información al respecto GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

Ejecute un escenario en un símbolo del sistema.

```
/// <summary>
/// Show Amazon Elastic Compute Cloud (Amazon EC2) Basics actions.
/// </summary>
public class EC2Basics
{
    /// <summary>
    /// Perform the actions defined for the Amazon EC2 Basics scenario.
    /// </summary>
```

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

    // Now the client is available for injection.
    var ec2Client = host.Services.GetRequiredService<IAmazonEC2>();
    var ec2Methods = new EC2Wrapper(ec2Client);

    var ssmClient =
host.Services.GetRequiredService<IAmazonSimpleSystemsManagement>();
    var ssmMethods = new SsmWrapper(ssmClient);
    var uiMethods = new UiMethods();

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

    // Start the scenario.
    uiMethods.DisplayOverview();
    uiMethods.PressEnter();

    // Create the key pair.
    uiMethods.DisplayTitle("Create RSA key pair");
    Console.WriteLine("Let's create an RSA key pair that you can use to ");
    Console.WriteLine("securely connect to your EC2 instance.");
    var keyPair = await ec2Methods.CreateKeyPair(keyPairName);

    // Save key pair information to a temporary file.
    var tempFileName = ec2Methods.SaveKeyPair(keyPair);
```

```
    Console.WriteLine($"Created the key pair: {keyPair.KeyName} and saved it
to: {tempFileName}");
    string? answer;
    do
    {
        Console.Write("Would you like to list your existing key pairs? ");
        answer = Console.ReadLine();
    } while (answer!.ToLower() != "y" && answer.ToLower() != "n");

    if (answer == "y")
    {
        // List existing key pairs.
        uiMethods.DisplayTitle("Existing key pairs");

        // Passing an empty string to the DescribeKeyPairs method will return
        // a list of all existing key pairs.
        var keyPairs = await ec2Methods.DescribeKeyPairs("");
        keyPairs.ForEach(kp =>
        {
            Console.WriteLine($"{kp.KeyName} created at: {kp.CreateTime}
Fingerprint: {kp.KeyFingerprint}");
        });
        uiMethods.PressEnter();

        // Create the security group.
        Console.WriteLine("Let's create a security group to manage access to your
instance.");
        var secGroupId = await ec2Methods.CreateSecurityGroup(groupName,
groupDescription);
        Console.WriteLine("Let's add rules to allow all HTTP and HTTPS inbound
traffic and to allow SSH only from your current IP address.");

        uiMethods.DisplayTitle("Security group information");
        var secGroups = await ec2Methods.DescribeSecurityGroups(secGroupId);

        Console.WriteLine($"Created security group {groupName} in your default
VPC.");
        secGroups.ForEach(group =>
        {
            ec2Methods.DisplaySecurityGroupInfoAsync(group);
        });
        uiMethods.PressEnter();
    }
}
```



```
    Console.WriteLine("Now we'll authorize the security group we just created
so that it can");
    Console.WriteLine("access the EC2 instances you create.");
    var success = await ec2Methods.AuthorizeSecurityGroupIngress(groupName);

    secGroups = await ec2Methods.DescribeSecurityGroups(secGroupId);
    Console.WriteLine($"Now let's look at the permissions again.");
    secGroups.ForEach(group =>
    {
        ec2Methods.DisplaySecurityGroupInfoAsync(group);
    });
    uiMethods.PressEnter();

    // Get list of available Amazon Linux 2 Amazon Machine Images (AMIs).
    var parameters = await ssmMethods.GetParametersByPath("/aws/service/ami-
amazon-linux-latest");

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

    var images = await ec2Methods.DescribeImages(imageIds);

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

    int choice;
    bool validNumber = false;

    do
    {
        Console.Write("Please select an image: ");
        var selImage = Console.ReadLine();
        validNumber = int.TryParse(selImage, out choice);
    } while (!validNumber);

    var selectedImage = images[choice - 1];

    // Display available instance types.
    uiMethods.DisplayTitle("Instance Types");
    var instanceTypes = await
ec2Methods.DescribeInstanceTypes(selectedImage.Architecture);
```

```
i = 1;
instanceTypes.ForEach(instanceType =>
{
    Console.WriteLine($"{i++}\t{instanceType.InstanceType}");
});

do
{
    Console.Write("Please select an instance type: ");
    var selImage = Console.ReadLine();
    validNumber = int.TryParse(selImage, out choice);
} while (!validNumber);

var selectedInstanceType = instanceTypes[choice - 1].InstanceType;

// Create an EC2 instance.
uiMethods.DisplayTitle("Creating an EC2 Instance");
var instanceId = await ec2Methods.RunInstances(selectedImage.ImageId,
selectedInstanceType, keyPairName, secGroupId);
Console.Write("Waiting for the instance to start.");
var isRunning = false;
do
{
    isRunning = await ec2Methods.WaitForInstanceState(instanceId,
InstanceStateName.Running);
} while (!isRunning);

uiMethods.PressEnter();

var instance = await ec2Methods.DescribeInstance(instanceId);
uiMethods.DisplayTitle("New Instance Information");
ec2Methods.DisplayInstanceInformation(instance);

Console.WriteLine("\nYou can use SSH to connect to your instance. For
example:");
Console.WriteLine($"{tssh -i {tempFileName} ec2-
user@{instance.PublicIpAddress}");

uiMethods.PressEnter();

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

await ec2Methods.StopInstances(instanceId);
```

```
var hasStopped = false;
do
{
    hasStopped = await ec2Methods.WaitForInstanceState(instanceId,
InstanceStateName.Stopped);
} while (!hasStopped);

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

Console.WriteLine("Now let's start it up again.");
await ec2Methods.StartInstances(instanceId);
Console.WriteLine("Waiting for instance to start. ");

isRunning = false;
do
{
    isRunning = await ec2Methods.WaitForInstanceState(instanceId,
InstanceStateName.Running);
} while (!isRunning);

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

instance = await ec2Methods.DescribeInstance(instanceId);
uiMethods.DisplayTitle("New Instance Information");
ec2Methods.DisplayInstanceInformation(instance);

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

uiMethods.PressEnter();

Console.WriteLine("Now we will stop the instance again. Then we will
create and associate an");
Console.WriteLine("Elastic IP address to use with our instance.");

await ec2Methods.StopInstances(instanceId);
hasStopped = false;
do
{
    hasStopped = await ec2Methods.WaitForInstanceState(instanceId,
InstanceStateName.Stopped);
} while (!hasStopped);
```

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

uiMethods.DisplayTitle("Allocate Elastic IP address");
Console.WriteLine("You can allocate an Elastic IP address and associate
it with your instance\nto keep a consistent IP address even when your instance
restarts.");
var allocationId = await ec2Methods.AllocateAddress();
Console.WriteLine("Now we will associate the Elastic IP address with our
instance.");
var associationId = await ec2Methods.AssociateAddress(allocationId,
instanceId);

// Start the instance again.
Console.WriteLine("Now let's start the instance again.");
await ec2Methods.StartInstances(instanceId);
Console.WriteLine("Waiting for instance to start. ");

isRunning = false;
do
{
    isRunning = await ec2Methods.WaitForInstanceState(instanceId,
InstanceStateName.Running);
} while (!isRunning);

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

instance = await ec2Methods.DescribeInstance(instanceId);
uiMethods.DisplayTitle("Instance information");
ec2Methods.DisplayInstanceInformation(instance);

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

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

await ec2Methods.StopInstances(instanceId);

hasStopped = false;
do
{
```

```
        hasStopped = await ec2Methods.WaitForInstanceState(instanceId,
InstanceStateName.Stopped);
    } while (!hasStopped);

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

    Console.WriteLine("Now let's start it up again.");
    await ec2Methods.StartInstances(instanceId);
    Console.WriteLine("Waiting for instance to start. ");

    isRunning = false;
    do
    {
        isRunning = await ec2Methods.WaitForInstanceState(instanceId,
InstanceStateName.Running);
    } while (!isRunning);

    instance = await ec2Methods.DescribeInstance(instanceId);
    uiMethods.DisplayTitle("New Instance Information");
    ec2Methods.DisplayInstanceInformation(instance);
    Console.WriteLine("Note that the IP address did not change this time.");
    uiMethods.PressEnter();

    uiMethods.DisplayTitle("Clean up resources");

    Console.WriteLine("Now let's clean up the resources we created.");

    // Terminate the instance.
    Console.WriteLine("Terminating the instance we created.");
    var stateChange = await ec2Methods.TerminateInstances(instanceId);

    // Wait for the instance state to be terminated.
    var hasTerminated = false;
    do
    {
        hasTerminated = await ec2Methods.WaitForInstanceState(instanceId,
InstanceStateName.Terminated);
    } while (!hasTerminated);

    Console.WriteLine($"The instance {instanceId} has been terminated.");
    Console.WriteLine("Now we can disassociate the Elastic IP address and
release it.");

    // Disassociate the Elastic IP address.
```

```

    var disassociated = ec2Methods.DisassociateIp(associationId);

    // Delete the Elastic IP address.
    var released = ec2Methods.ReleaseAddress(allocationId);

    // Delete the security group.
    Console.WriteLine($"Deleting the Security Group: {groupName}.");
    success = await ec2Methods.DeleteSecurityGroup(secGroupId);
    if (success)
    {
        Console.WriteLine($"Successfully deleted {groupName}.");
    }

    // Delete the RSA key pair.
    Console.WriteLine($"Deleting the key pair: {keyPairName}");
    await ec2Methods.DeleteKeyPair(keyPairName);
    Console.WriteLine("Deleting the temporary file with the key
information.");
    ec2Methods.DeleteTempFile(tempFileName);
    uiMethods.PressEnter();

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

```

Defina una clase que ajuste las acciones de EC2.

```

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

    public EC2Wrapper(IAmazonEC2 amazonService)
    {
        _amazonEC2 = amazonService;
    }

    /// <summary>
    /// Allocate an Elastic IP address.

```

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

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

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

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

/// <summary>
/// Authorize the local computer ingress to EC2 instances associated
/// with the virtual private cloud (VPC) security group.
/// </summary>
/// <param name="groupName">The name of the security group.</param>
/// <returns>A Boolean value indicating the success of the action.</returns>
public async Task<bool> AuthorizeSecurityGroupIngress(string groupName)
{
    // Get the IP address for the local computer.
    var ipAddress = await GetIpAddress();
    Console.WriteLine($"Your IP address is: {ipAddress}");
}
```

```

        var ipRanges = new List<IpRange> { new IpRange { CidrIp =
"${ipAddress}/32" } };
        var permission = new IpPermission
        {
            Ipv4Ranges = ipRanges,
            IpProtocol = "tcp",
            FromPort = 22,
            ToPort = 22
        };
        var permissions = new List<IpPermission> { permission };
        var response = await _amazonEC2.AuthorizeSecurityGroupIngressAsync(
            new AuthorizeSecurityGroupIngressRequest(groupName, permissions));
        return response.HttpStatusCode == HttpStatusCode.OK;
    }

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

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

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

```



```

    var response = await _amazonEC2.CreateKeyPairAsync(request);

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

/// <summary>
/// Save KeyPair information to a temporary file.
/// </summary>
/// <param name="keyPair">The name of the key pair.</param>
/// <returns>The full path to the temporary file.</returns>
public string SaveKeyPair(KeyPair keyPair)
{
    var tempPath = Path.GetTempPath();
    var tempFileName = $"{tempPath}\\{Path.GetRandomFileName()}";
    var pemFileName = Path.ChangeExtension(tempFileName, "pem");

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

    return pemFileName;
}

/// <summary>
/// Create an Amazon EC2 security group.
/// </summary>
/// <param name="groupName">The name for the new security group.</param>
/// <param name="groupDescription">A description of the new security group.</
param>
/// <returns>The group Id of the new security group.</returns>
public async Task<string> CreateSecurityGroup(string groupName, string
groupDescription)
{
    var response = await _amazonEC2.CreateSecurityGroupAsync(

```

```

        new CreateSecurityGroupRequest(groupName, groupDescription));

    return response.GroupId;
}

/// <summary>
/// Create a new Amazon EC2 VPC.
/// </summary>
/// <param name="cidrBlock">The CIDR block for the new security group.</
param>
/// <returns>The VPC Id of the new VPC.</returns>
public async Task<string?> CreateVPC(string cidrBlock)
{
    try
    {
        var response = await _amazonEC2.CreateVpcAsync(new CreateVpcRequest
        {
            CidrBlock = cidrBlock,
        });

        Vpc vpc = response.Vpc;
        Console.WriteLine($"Created VPC with ID: {vpc.VpcId}.");
        return vpc.VpcId;
    }
    catch (AmazonEC2Exception ex)
    {
        Console.WriteLine($"Couldn't create VPC because: {ex.Message}");
        return null;
    }
}

/// <summary>
/// Delete an Amazon EC2 key pair.
/// </summary>
/// <param name="keyPairName">The name of the key pair to delete.</param>
/// <returns>A Boolean value indicating the success of the action.</returns>
public async Task<bool> DeleteKeyPair(string keyPairName)
{
    try
    {
        await _amazonEC2.DeleteKeyPairAsync(new
DeleteKeyPairRequest(keyPairName)).ConfigureAwait(false);

```

```
        return true;
    }
    catch (Exception ex)
    {
        Console.WriteLine($"Couldn't delete the key pair because:
{ex.Message}");
        return false;
    }
}

/// <summary>
/// Delete the temporary file where the key pair information was saved.
/// </summary>
/// <param name="tempFileName">The path to the temporary file.</param>
public void DeleteTempFile(string tempFileName)
{
    if (File.Exists(tempFileName))
    {
        File.Delete(tempFileName);
    }
}

/// <summary>
/// Delete an Amazon EC2 security group.
/// </summary>
/// <param name="groupName">The name of the group to delete.</param>
/// <returns>A Boolean value indicating the success of the action.</returns>
public async Task<bool> DeleteSecurityGroup(string groupId)
{
    var response = await _amazonEC2.DeleteSecurityGroupAsync(new
DeleteSecurityGroupRequest { GroupId = groupId });
    return response.HttpStatusCode == HttpStatusCode.OK;
}

/// <summary>
/// Delete an Amazon EC2 VPC.
/// </summary>
/// <returns>A Boolean value indicating the success of the action.</returns>
public async Task<bool> DeleteVpc(string vpcId)
{
    var request = new DeleteVpcRequest
    {
        VpcId = vpcId,
    };
};
```

```
        var response = await _amazonEC2.DeleteVpcAsync(request);

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }

    /// <summary>
    /// Get information about existing Amazon EC2 images.
    /// </summary>
    /// <returns>A list of image information.</returns>
    public async Task<List<Image>> DescribeImages(List<string>? imageIds)
    {
        var request = new DescribeImagesRequest();
        if (imageIds is not null)
        {
            // If the imageIds list is not null, add the list
            // to the request object.
            request.ImageIds = imageIds;
        }

        var response = await _amazonEC2.DescribeImagesAsync(request);
        return response.Images;
    }

    /// <summary>
    /// Display the information returned by DescribeImages.
    /// </summary>
    /// <param name="images">The list of image information to display.</param>
    public void DisplayImageInfo(List<Image> images)
    {
        images.ForEach(image =>
        {
            Console.WriteLine($"{image.Name} Created on: {image.CreationDate}");
        });
    }

    /// <summary>
    /// Get information about an Amazon EC2 instance.
    /// </summary>
    /// <param name="instanceId">The instance Id of the EC2 instance.</param>
    /// <returns>An EC2 instance.</returns>
    public async Task<Instance> DescribeInstance(string instanceId)
    {
```

```
        var response = await _amazonEC2.DescribeInstancesAsync(
            new DescribeInstancesRequest { InstanceIds = new List<string>
{ instanceId } });
        return response.Reservations[0].Instances[0];
    }

    /// <summary>
    /// Display EC2 instance information.
    /// </summary>
    /// <param name="instance">The instance Id of the EC2 instance.</param>
    public void DisplayInstanceInformation(Instance instance)
    {
        Console.WriteLine($"ID: {instance.InstanceId}");
        Console.WriteLine($"Image ID: {instance.ImageId}");
        Console.WriteLine($"InstanceType: {instance.InstanceType}");
        Console.WriteLine($"Key Name: {instance.KeyName}");
        Console.WriteLine($"VPC ID: {instance.VpcId}");
        Console.WriteLine($"Public IP: {instance.PublicIpAddress}");
        Console.WriteLine($"State: {instance.State.Name}");
    }

    /// <summary>
    /// Get information about existing EC2 images.
    /// </summary>
    /// <returns>Async task.</returns>
    public async Task DescribeInstances()
    {
        // List all EC2 instances.
        await GetInstanceDescriptions();

        string tagName = "IncludeInList";
        string tagValue = "Yes";
        await GetInstanceDescriptionsFiltered(tagName, tagValue);
    }

    /// <summary>
    /// Get information for all existing Amazon EC2 instances.
    /// </summary>
    /// <returns>Async task.</returns>
    public async Task GetInstanceDescriptions()
    {
        Console.WriteLine("Showing all instances:");
        var paginator = _amazonEC2.Paginators.DescribeInstances(new
DescribeInstancesRequest());
```

```
await foreach (var response in paginator.Responses)
{
    foreach (var reservation in response.Reservations)
    {
        foreach (var instance in reservation.Instances)
        {
            Console.WriteLine($"Instance ID: {instance.InstanceId}");
            Console.WriteLine($"\\tCurrent State: {instance.State.Name}");
        }
    }
}

/// <summary>
/// Get information about EC2 instances filtered by a tag name and value.
/// </summary>
/// <param name="tagName">The name of the tag to filter on.</param>
/// <param name="tagValue">The value of the tag to look for.</param>
/// <returns>Async task.</returns>
public async Task GetInstanceDescriptionsFiltered(string tagName, string
tagValue)
{
    // This tag filters the results of the instance list.
    var filters = new List<Filter>
    {
        new Filter
        {
            Name = $"tag:{tagName}",
            Values = new List<string>
            {
                tagValue,
            },
        },
    };
    var request = new DescribeInstancesRequest
    {
        Filters = filters,
    };

    Console.WriteLine("\\nShowing instances with tag: \\\"IncludeInList\\\" set to
\\\"Yes\\\".");
    var paginator = _amazonEC2.Paginators.DescribeInstances(request);
```

```

        await foreach (var response in paginator.Responses)
        {
            foreach (var reservation in response.Reservations)
            {
                foreach (var instance in reservation.Instances)
                {
                    Console.WriteLine($"Instance ID: {instance.InstanceId} ");
                    Console.WriteLine($"\\tCurrent State: {instance.State.Name}");
                }
            }
        }
    }

    /// <summary>
    /// Describe the instance types available.
    /// </summary>
    /// <returns>A list of instance type information.</returns>
    public async Task<List<InstanceTypeInfo>>
DescribeInstanceTypes(ArchitectureValues architecture)
    {
        var request = new DescribeInstanceTypesRequest();

        var filters = new List<Filter>
            { new Filter("processor-info.supported-architecture", new
List<string> { architecture.ToString() }) };
        filters.Add(new Filter("instance-type", new() { "*.micro", "*.small" }));

        request.Filters = filters;
        var instanceTypes = new List<InstanceTypeInfo>();

        var paginator = _amazonEC2.Paginators.DescribeInstanceTypes(request);
        await foreach (var instanceType in paginator.InstanceTypes)
        {
            instanceTypes.Add(instanceType);
        }
        return instanceTypes;
    }

    /// <summary>
    /// Display the instance type information returned by
DescribeInstanceTypesAsync.
    /// </summary>
    /// <param name="instanceTypes">The list of instance type information.</
param>

```

```
public void DisplayInstanceTypeInfo(List<InstanceTypeInfo> instanceTypes)
{
    instanceTypes.ForEach(type =>
    {
        Console.WriteLine($"{type.InstanceType}\t{type.MemoryInfo}");
    });
}

/// <summary>
/// Get information about an Amazon EC2 key pair.
/// </summary>
/// <param name="keyPairName">The name of the key pair.</param>
/// <returns>A list of key pair information.</returns>
public async Task<List<KeyPairInfo>> DescribeKeyPairs(string keyPairName)
{
    var request = new DescribeKeyPairsRequest();
    if (!string.IsNullOrEmpty(keyPairName))
    {
        request = new DescribeKeyPairsRequest
        {
            KeyNames = new List<string> { keyPairName }
        };
    }
    var response = await _amazonEC2.DescribeKeyPairsAsync(request);
    return response.KeyPairs.ToList();
}

/// <summary>
/// Retrieve information for an Amazon EC2 security group.
/// </summary>
/// <param name="groupId">The Id of the Amazon EC2 security group.</param>
/// <returns>A list of security group information.</returns>
public async Task<List<SecurityGroup>> DescribeSecurityGroups(string groupId)
{
    var request = new DescribeSecurityGroupsRequest();
    var groupIds = new List<string> { groupId };
    request.GroupIds = groupIds;

    var response = await _amazonEC2.DescribeSecurityGroupsAsync(request);
    return response.SecurityGroups;
}

/// <summary>
```



```
/// Display the information returned by the call to
/// DescribeSecurityGroupsAsync.
/// </summary>
/// <param name="securityGroup">A list of security group information.</param>
public void DisplaySecurityGroupInfoAsync(SecurityGroup securityGroup)
{
    Console.WriteLine($"{securityGroup.GroupName}");
    Console.WriteLine("Ingress permissions:");
    securityGroup.IpPermissions.ForEach(permission =>
    {
        Console.WriteLine($"  \tFromPort: {permission.FromPort}");
        Console.WriteLine($"  \tIpProtocol: {permission.IpProtocol}");

        Console.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)
{
    var response = await _amazonEC2.DisassociateAddressAsync(
        new DisassociateAddressRequest { AssociationId = associationId });
    return response.HttpStatusCode == HttpStatusCode.OK;
}

/// <summary>
/// Retrieve a list of available Amazon Linux images.
/// </summary>
/// <returns>A list of image information.</returns>
public async Task<List<Image>> GetEC2AmiList()
{
    var filter = new Filter { Name = "architecture", Values = new
List<string> { "x86_64" } };
    var filters = new List<Filter> { filter };
    var response = await _amazonEC2.DescribeImagesAsync(new
DescribeImagesRequest { Filters = filters });
    return response.Images;
}

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

    var response = await _amazonEC2.RebootInstancesAsync(request);
}
```

```
        if (response.HttpStatusCode == System.Net.HttpStatusCode.OK)
        {
            Console.WriteLine("Instances successfully rebooted.");
        }
        else
        {
            Console.WriteLine("Could not reboot one or more instances.");
        }
    }

    /// <summary>
    /// Release an Elastic IP address.
    /// </summary>
    /// <param name="allocationId">The allocation Id of the Elastic IP address.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> ReleaseAddress(string allocationId)
    {
        var request = new ReleaseAddressRequest
        {
            AllocationId = allocationId
        };

        var response = await _amazonEC2.ReleaseAddressAsync(request);
        return response.HttpStatusCode == HttpStatusCode.OK;
    }

    /// <summary>
    /// Create and run an EC2 instance.
    /// </summary>
    /// <param name="ImageId">The image Id of the image used as a basis for the
    /// EC2 instance.</param>
    /// <param name="instanceType">The instance type of the EC2 instance to
    create.</param>
    /// <param name="keyName">The name of the key pair to associate with the
    /// instance.</param>
    /// <param name="groupId">The Id of the Amazon EC2 security group that will
    be
    /// allowed to interact with the new EC2 instance.</param>
    /// <returns>The instance Id of the new EC2 instance.</returns>
    public async Task<string> RunInstances(string imageId, string instanceType,
    string keyName, string groupId)
    {
        var request = new RunInstancesRequest
```

```
    {
        ImageId = imageId,
        InstanceType = instanceType,
        KeyName = keyName,
        MinCount = 1,
        MaxCount = 1,
        SecurityGroupIds = new List<string> { groupId }
    };
    var response = await _amazonEC2.RunInstancesAsync(request);
    return response.Reservation.Instances[0].InstanceId;
}

/// <summary>
/// Start an EC2 instance.
/// </summary>
/// <param name="ec2InstanceId">The instance Id of the Amazon EC2 instance
/// to start.</param>
/// <returns>Async task.</returns>
public async Task StartInstances(string ec2InstanceId)
{
    var request = new StartInstancesRequest
    {
        InstanceIds = new List<string> { ec2InstanceId },
    };

    var response = await _amazonEC2.StartInstancesAsync(request);

    if (response.StartingInstances.Count > 0)
    {
        var instances = response.StartingInstances;
        instances.ForEach(i =>
        {
            Console.WriteLine($"Successfully started the EC2 instance with
instance ID: {i.InstanceId}.");
        });
    }
}

/// <summary>
/// Stop an EC2 instance.
/// </summary>
/// <param name="ec2InstanceId">The instance Id of the EC2 instance to
/// stop.</param>
```

```
/// <returns>Async task.</returns>
public async Task StopInstances(string ec2InstanceId)
{
    // In addition to the list of instance Ids, the
    // request can also include the following properties:
    //     Force      When true, forces the instances to
    //                 stop but you must check the integrity
    //                 of the file system. Not recommended on
    //                 Windows instances.
    //     Hibernate  When true, hibernates the instance if the
    //                 instance was enabled for hibernation when
    //                 it was launched.
    var request = new StopInstancesRequest
    {
        InstanceIds = new List<string> { ec2InstanceId },
    };

    var response = await _amazonEC2.StopInstancesAsync(request);

    if (response.StoppingInstances.Count > 0)
    {
        var instances = response.StoppingInstances;
        instances.ForEach(i =>
        {
            Console.WriteLine($"Successfully stopped the EC2 Instance " +
                              $"with InstanceID: {i.InstanceId}.");
        });
    }
}

/// <summary>
/// Terminate an EC2 instance.
/// </summary>
/// <param name="ec2InstanceId">The instance Id of the EC2 instance
/// to terminate.</param>
/// <returns>Async task.</returns>
public async Task<List<InstanceStateChange>> TerminateInstances(string
ec2InstanceId)
{
    var request = new TerminateInstancesRequest
    {
        InstanceIds = new List<string> { ec2InstanceId }
    };
};
```

```

        var response = await _amazonEC2.TerminateInstancesAsync(request);
        return response.TerminatingInstances;
    }

    /// <summary>
    /// Wait until an EC2 instance is in a specified state.
    /// </summary>
    /// <param name="instanceId">The instance Id.</param>
    /// <param name="stateName">The state to wait for.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> WaitForInstanceState(string instanceId,
InstanceStateName stateName)
    {
        var request = new DescribeInstancesRequest
        {
            InstanceIds = new List<string> { instanceId }
        };

        // Wait until the instance is running.
        var hasState = false;
        do
        {
            // Wait 5 seconds.
            Thread.Sleep(5000);

            // Check for the desired state.
            var response = await _amazonEC2.DescribeInstancesAsync(request);
            var instance = response.Reservations[0].Instances[0];
            hasState = instance.State.Name == stateName;
            Console.WriteLine(". ");
        } while (!hasState);

        return hasState;
    }
}

```

- Para obtener información sobre la API, consulte los siguientes temas en la referencia de la API de AWS SDK for .NET .
  - [AllocateAddress](#)
  - [AssociateAddress](#)

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

## Bash

### AWS CLI con el script Bash

#### Note

Hay más información. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

Ejecutar un escenario interactivo en un símbolo del sistema.

```
#####  
# 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
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++))
        fi
    fi
}

```

```

    echo -n "$get_input_result"
else
    echo "MOCK_INPUT_ARRAY has no more elements" 1>&2
    return 1
fi
fi

return 0
}

#####
# function yes_no_input
#
# This function requests a yes/no answer from the user, following to a prompt.
#
# Parameters:
#     $1 - The prompt.
#
# Returns:
#     0 - If yes.
#     1 - If no.
#####
function yes_no_input() {
    if [ -z "$1" ]; then
        echo "Internal error yes_no_input"
        return 1
    fi

    local index=0
    local response="N"
    while [[ $index -lt 10 ]]; do
        index=$((index + 1))
        echo -n "$1"
        if ! get_input; then
            return 1
        fi
        response=$(echo "$get_input_result" | tr '[:upper:]' '[:lower:]')
        if [ "$response" = "y" ] || [ "$response" = "n" ]; then
            break
        else
            echo -e "\nPlease enter or 'y' or 'n'."
        fi
    done
}

```

```

echo

if [ "$response" = "y" ]; then
    return 0
else
    return 1
fi
}

#####
# function integer_input
#
# This function prompts the user to enter an integer within a specified range
# and validates the input.
#
# Parameters:
#     $1 - The prompt message to display to the user.
#     $2 - The minimum value of the accepted range.
#     $3 - The maximum value of the accepted range.
#
# Returns:
#     The valid integer input from the user.
#     If the input is invalid or out of range, the function will continue
#     prompting the user until a valid input is provided.
#####
function integer_input() {
    local prompt="$1"
    local min_value="$2"
    local max_value="$3"
    local input=""

    while true; do
        # Display the prompt message and wait for user input
        echo -n "$prompt"

        if ! get_input; then
            return 1
        fi

        input="$get_input_result"

        # Check if the input is a valid integer
        if [[ "$input" =~ ^-[0-9]+$ ]]; then
            # Check if the input is within the specified range

```

```

    if ((input >= min_value && input <= max_value)); then
        return 0
    else
        echo "Error: Input, $input, must be between $min_value and $max_value."
    fi
else
    echo "Error: Invalid input- $input. Please enter an integer."
fi
done
}
#####
# function new_line_and_tab_to_list
#
# This function takes a string input containing newlines and tabs, and
# converts it into a list (array) of elements.
#
# Parameters:
#     $1 - The input string containing newlines and tabs.
#
# Returns:
#     The resulting list (array) is stored in the global variable
#     'list_result'.
#####
function new_line_and_tab_to_list() {
    local input=$1
    export list_result

    list_result=()
    mapfile -t lines <<<"$input"
    local line
    for line in "${lines[@]"; do
        IFS=$'\t' read -ra parameters <<<"$line"
        list_result+=("${parameters[@]}")
    done
}

#####
# function echo_repeat
#
# This function prints a string 'n' times to stdout.
#
# Parameters:
#     $1 - The string.
#     $2 - Number of times to print the string.

```



```

#
# Outputs:
#   String 'n' times to stdout.
#
# Returns:
#   0
#####
function echo_repeat() {
    local end=$2
    for ((i = 0; i < end; i++)); do
        echo -n "$1"
    done
    echo
}

```

Las funciones de DynamoDB utilizadas en este escenario.

```

#####
# 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
}
```

```

    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
        esac
    done
}

```

```

        ;;
    \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
done
export OPTIND=1

local query="SecurityGroups[*].[GroupName, GroupId, VpcId, IpPermissions[*].
[IpProtocol, FromPort, ToPort, IpRanges[*].CidrIp]]"

if [[ -n "$security_group_id" ]]; then
    response=$(aws ec2 describe-security-groups --group-ids "$security_group_id"
--query "${query}" --output text)
else
    response=$(aws ec2 describe-security-groups --query "${query}" --output text)
fi

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports describe-security-groups operation failed.
$response"
    return 1
fi

echo "$response"

return 0
}

#####
# function ec2_authorize_security_group_ingress
#
# This function authorizes an ingress rule for an Amazon Elastic Compute Cloud
(Amazon EC2) security group.
#
# Parameters:
#     -g security_group_id - The ID of the security group.
#     -i ip_address - The IP address or CIDR block to authorize.
#     -p protocol - The protocol to authorize (e.g., tcp, udp, icmp).

```

```

# -f from_port - The start of the port range to authorize.
# -t to_port - The end of the port range to authorize.
#
# And:
# 0 - If successful.
# 1 - If it fails.
#####
function ec2_authorize_security_group_ingress() {
    local security_group_id ip_address protocol from_port to_port response
    local option OPTARG # Required to use 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

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
}

```

```

    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}" ;;
  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
        ;;
    \?)
        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 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

    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
                ;;
        esac
    done
}

```

```

    \?)
    echo "Invalid parameter"
    usage
    return 1
    ;;
esac
done
export OPTIND=1

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

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

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

echo "$response"
return 0
}

#####
# function ec2_associate_address
#
# This function associates an Elastic IP address with an Amazon Elastic Compute
Cloud (Amazon EC2) instance.
#
# Parameters:

```

```

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

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

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

    # Validate the input parameters

```

```

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

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

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

echo "$response"
return 0
}

#####
# function ec2_disassociate_address
#
# This function disassociates an Elastic IP address from an Amazon Elastic
# Compute Cloud (Amazon EC2) instance.
#
# Parameters:
#     -a association_id - The association ID that represents the association of
#     the Elastic IP address with an instance.
#
# And:
#     0 - If successful.
#     1 - If it fails.
#
#####
function ec2_disassociate_address() {
    local association_id response

```



```
# Function to display usage information
function usage() {
    echo "function ec2_disassociate_address"
    echo "Disassociates an Elastic IP address from an Amazon Elastic Compute
Cloud (Amazon EC2) instance."
    echo " -a association_id - The association ID that represents the
association of the Elastic IP address with an instance."
    echo ""
}

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

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

response=$(aws ec2 disassociate-address \
    --association-id "$association_id") || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports disassociate-address operation failed."
    errecho "$response"
    return 1
}

return 0
}
```

```
#####  
# function ec2_release_address  
#  
# This function releases an Elastic IP address from an Amazon Elastic Compute  
# Cloud (Amazon EC2) instance.  
#  
# Parameters:  
#     -a allocation_id - The allocation ID of the Elastic IP address to  
#     release.  
#  
# Returns:  
#     0 - If successful.  
#     1 - If it fails.  
#  
#####  
function ec2_release_address() {  
    local allocation_id response  
  
    # Function to display usage information  
    function usage() {  
        echo "function ec2_release_address"  
        echo "Releases an Elastic IP address from an Amazon Elastic Compute Cloud  
(Amazon EC2) instance."  
        echo "  -a allocation_id - The allocation ID of the Elastic IP address to  
release."  
        echo ""  
    }  
  
    # Parse the command-line arguments  
    while getopts "a:h" option; do  
        case "${option}" in  
            a) allocation_id="${OPTARG}" ;;  
            h)  
                usage  
                return 0  
                ;;  
            \?)  
                echo "Invalid parameter"  
                usage  
                return 1  
                ;;  
        esac  
    done  
    export OPTIND=1
```

```

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

response=$(aws ec2 release-address \
    --allocation-id "$allocation_id") || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports release-address operation failed."
    errecho "$response"
    return 1
}

return 0
}

#####
# function ec2_terminate_instances
#
# This function terminates one or more Amazon Elastic Compute Cloud (Amazon EC2)
# instances using the AWS CLI.
#
# Parameters:
#     -i instance_ids - A space-separated list of instance IDs.
#     -h - Display help.
#
# Returns:
#     0 - If successful.
#     1 - If it fails.
#####
function ec2_terminate_instances() {
    local instance_ids response
    local option OPTARG # Required to use 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
}

#####
# 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
}

```

Las funciones de utilidad usadas en este escenario.

```

#####
# 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 obtener información sobre la API, consulte los siguientes temas en la Referencia de comandos de AWS CLI .
  - [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

Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

```
/**
 * Before running this Java (v2) code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
 * started.html
 *
 * This Java example performs the following tasks:
 *
 * 1. Creates an RSA key pair and saves the private key data as a .pem file.
 * 2. Lists key pairs.
 * 3. Creates a security group for the default VPC.
 * 4. Displays security group information.
 * 5. Gets a list of Amazon Linux 2 AMIs and selects one.
 * 6. Gets more information about the image.
 * 7. Gets a list of instance types that are compatible with the selected AMI's
```

```

* architecture.
* 8. Creates an instance with the key pair, security group, AMI, and an
* instance type.
* 9. Displays information about the instance.
* 10. Stops the instance and waits for it to stop.
* 11. Starts the instance and waits for it to start.
* 12. Allocates an Elastic IP address and associates it with the instance.
* 13. Displays SSH connection info for the instance.
* 14. Disassociates and deletes the Elastic IP address.
* 15. Terminates the instance and waits for it to terminate.
* 16. Deletes the security group.
* 17. Deletes the key pair.
*/
public class EC2Scenario {
    public static final String DASHES = new String(new char[80]).replace("\0",
"-");

    public static void main(String[] args) throws InterruptedException {

        final String usage = ""

            Usage:
                <keyName> <fileName> <groupName> <groupDesc> <vpcId>

            Where:
                keyName - A key pair name (for example, TestKeyPair).\s
                fileName - A file name where the key information is written
to.\s

                groupName - The name of the security group.\s
                groupDesc - The description of the security group.\s
                vpcId - A VPC Id value. You can get this value from the AWS
Management Console.\s
                myIpAddress - The IP address of your development machine.\s

            """;

        if (args.length != 6) {
            System.out.println(usage);
            System.exit(1);
        }

        String keyName = args[0];
        String fileName = args[1];
        String groupName = args[2];

```

```
String groupDesc = args[3];
String vpcId = args[4];
String myIpAddress = args[5];

Region region = Region.US_WEST_2;
Ec2Client ec2 = Ec2Client.builder()
    .region(region)
    .build();

SsmClient ssmClient = SsmClient.builder()
    .region(region)
    .build();

System.out.println(DASHES);
System.out.println("Welcome to the Amazon EC2 example scenario.");
System.out.println(DASHES);

System.out.println(DASHES);
System.out.println("1. Create an RSA key pair and save the private key
material as a .pem file.");
createKeyPair(ec2, keyName, fileName);
System.out.println(DASHES);

System.out.println(DASHES);
System.out.println("2. List key pairs.");
describeKeys(ec2);
System.out.println(DASHES);

System.out.println(DASHES);
System.out.println("3. Create a security group.");
String groupId = createSecurityGroup(ec2, groupName, groupDesc, vpcId,
myIpAddress);
System.out.println(DASHES);

System.out.println(DASHES);
System.out.println("4. Display security group info for the newly created
security group.");
describeSecurityGroups(ec2, groupId);
System.out.println(DASHES);

System.out.println(DASHES);
System.out.println("5. Get a list of Amazon Linux 2 AMIs and selects one
with amzn2 in the name.");
String instanceId = getParaValues(ssmClient);
```

```
System.out.println("The instance Id is " + instanceId);
System.out.println(DASHES);

System.out.println(DASHES);
System.out.println("6. Get more information about an amzn2 image.");
String amiValue = describeImage(ec2, instanceId);
System.out.println(DASHES);

System.out.println(DASHES);
System.out.println("7. Get a list of instance types.");
String instanceType = getInstanceTypes(ec2);
System.out.println("The instance type is " + instanceType);
System.out.println(DASHES);

System.out.println(DASHES);
System.out.println("8. Create an instance.");
String newInstanceId = runInstance(ec2, instanceType, keyName, groupName,
amiValue);
System.out.println("The instance Id is " + newInstanceId);
System.out.println(DASHES);

System.out.println(DASHES);
System.out.println("9. Display information about the running instance.
");
String ipAddress = describeEC2Instances(ec2, newInstanceId);
System.out.println("You can SSH to the instance using this command:");
System.out.println("ssh -i " + fileName + "ec2-user@" + ipAddress);
System.out.println(DASHES);

System.out.println(DASHES);
System.out.println("10. Stop the instance and use a waiter.");
stopInstance(ec2, newInstanceId);
System.out.println(DASHES);

System.out.println(DASHES);
System.out.println("11. Start the instance and use a waiter.");
startInstance(ec2, newInstanceId);
ipAddress = describeEC2Instances(ec2, newInstanceId);
System.out.println("You can SSH to the instance using this command:");
System.out.println("ssh -i " + fileName + "ec2-user@" + ipAddress);
System.out.println(DASHES);

System.out.println(DASHES);
```

```
        System.out.println("12. Allocate an Elastic IP address and associate it
with the instance.");
        String allocationId = allocateAddress(ec2);
        System.out.println("The allocation Id value is " + allocationId);
        String associationId = associateAddress(ec2, newInstanceId,
allocationId);
        System.out.println("The associate Id value is " + associationId);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("13. Describe the instance again.");
        ipAddress = describeEC2Instances(ec2, newInstanceId);
        System.out.println("You can SSH to the instance using this command:");
        System.out.println("ssh -i " + fileName + "ec2-user@" + ipAddress);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("14. Disassociate and release the Elastic IP
address.");
        disassociateAddress(ec2, associationId);
        releaseEC2Address(ec2, allocationId);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("15. Terminate the instance and use a waiter.");
        terminateEC2(ec2, newInstanceId);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("16. Delete the security group.");
        deleteEC2SecGroup(ec2, groupId);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("17. Delete the key.");
        deleteKeys(ec2, keyName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("You successfully completed the Amazon EC2
scenario.");
        System.out.println(DASHES);
        ec2.close();
    }
```

```
public static void deleteEC2SecGroup(Ec2Client ec2, String groupId) {
    try {
        DeleteSecurityGroupRequest request =
DeleteSecurityGroupRequest.builder()
            .groupId(groupId)
            .build();

        ec2.deleteSecurityGroup(request);
        System.out.println("Successfully deleted security group with Id " +
groupId);

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

public static void terminateEC2(Ec2Client ec2, String instanceId) {
    try {
        Ec2Waiter ec2Waiter = Ec2Waiter.builder()
            .overrideConfiguration(b -> b.maxAttempts(100))
            .client(ec2)
            .build();

        TerminateInstancesRequest ti = TerminateInstancesRequest.builder()
            .instanceIds(instanceId)
            .build();

        System.out.println("Use an Ec2Waiter to wait for the instance to
terminate. This will take a few minutes.");
        ec2.terminateInstances(ti);
        DescribeInstancesRequest instanceRequest =
DescribeInstancesRequest.builder()
            .instanceIds(instanceId)
            .build();

        WaiterResponse<DescribeInstancesResponse> waiterResponse = ec2Waiter
            .waitUntilInstanceTerminated(instanceRequest);
        waiterResponse.matched().response().ifPresent(System.out::println);
        System.out.println("Successfully started instance " + instanceId);
        System.out.println(instanceId + " is terminated!");

    } catch (Ec2Exception e) {
```

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

public static void deleteKeys(Ec2Client ec2, String keyPair) {
    try {
        DeleteKeyPairRequest request = DeleteKeyPairRequest.builder()
            .keyName(keyPair)
            .build();

        ec2.deleteKeyPair(request);
        System.out.println("Successfully deleted key pair named " + keyPair);

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

public static void releaseEC2Address(Ec2Client ec2, String allocId) {
    try {
        ReleaseAddressRequest request = ReleaseAddressRequest.builder()
            .allocationId(allocId)
            .build();

        ec2.releaseAddress(request);
        System.out.println("Successfully released Elastic IP address " +
allocId);
    } catch (Ec2Exception e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
}

public static void disassociateAddress(Ec2Client ec2, String associationId) {
    try {
        DisassociateAddressRequest addressRequest =
DisassociateAddressRequest.builder()
            .associationId(associationId)
            .build();

        ec2.disassociateAddress(addressRequest);
        System.out.println("You successfully disassociated the address!");
    }
}
```

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

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

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

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

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

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

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



```
public static void startInstance(Ec2Client ec2, String instanceId) {
    Ec2Waiter ec2Waiter = Ec2Waiter.builder()
        .overrideConfiguration(b -> b.maxAttempts(100))
        .client(ec2)
        .build();

    StartInstancesRequest request = StartInstancesRequest.builder()
        .instanceIds(instanceId)
        .build();

    System.out.println("Use an Ec2Waiter to wait for the instance to run.
This will take a few minutes.");
    ec2.startInstances(request);
    DescribeInstancesRequest instanceRequest =
DescribeInstancesRequest.builder()
    .instanceIds(instanceId)
    .build();

    WaiterResponse<DescribeInstancesResponse> waiterResponse =
ec2Waiter.waitUntilInstanceRunning(instanceRequest);
    waiterResponse.matched().response().ifPresent(System.out::println);
    System.out.println("Successfully started instance " + instanceId);
}

public static void stopInstance(Ec2Client ec2, String instanceId) {
    Ec2Waiter ec2Waiter = Ec2Waiter.builder()
        .overrideConfiguration(b -> b.maxAttempts(100))
        .client(ec2)
        .build();

    StopInstancesRequest request = StopInstancesRequest.builder()
        .instanceIds(instanceId)
        .build();

    System.out.println("Use an Ec2Waiter to wait for the instance to stop.
This will take a few minutes.");
    ec2.stopInstances(request);
    DescribeInstancesRequest instanceRequest =
DescribeInstancesRequest.builder()
    .instanceIds(instanceId)
    .build();

    WaiterResponse<DescribeInstancesResponse> waiterResponse =
ec2Waiter.waitUntilInstanceStopped(instanceRequest);
```

```
        waiterResponse.matched().response().ifPresent(System.out::println);
        System.out.println("Successfully stopped instance " + instanceId);
    }

    public static String describeEC2Instances(Ec2Client ec2, String
newInstanceId) {
        try {
            String pubAddress = "";
            boolean isRunning = false;
            DescribeInstancesRequest request = DescribeInstancesRequest.builder()
                .instanceIds(newInstanceId)
                .build();

            while (!isRunning) {
                DescribeInstancesResponse response =
ec2.describeInstances(request);
                String state =
response.reservations().get(0).instances().get(0).state().name().name();
                if (state.compareTo("RUNNING") == 0) {
                    System.out.println("Image id is " +
response.reservations().get(0).instances().get(0).imageId());
                    System.out.println(
                        "Instance type is " +
response.reservations().get(0).instances().get(0).instanceType());
                    System.out.println(
                        "Instance state is " +
response.reservations().get(0).instances().get(0).state().name());
                    pubAddress =
response.reservations().get(0).instances().get(0).publicIpAddress();
                    System.out.println("Instance address is " + pubAddress);
                    isRunning = true;
                }
            }
            return pubAddress;
        } catch (SsmException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
        return "";
    }

    public static String runInstance(Ec2Client ec2, String instanceType, String
keyName, String groupName,
        String amiId) {
```

```
    try {
        RunInstancesRequest runRequest = RunInstancesRequest.builder()
            .instanceType(instanceType)
            .keyName(keyName)
            .securityGroups(groupName)
            .maxCount(1)
            .minCount(1)
            .imageId(amiId)
            .build();

        System.out.println("Going to start an EC2 instance using a waiter");
        RunInstancesResponse response = ec2.runInstances(runRequest);
        String instanceIdVal = response.instances().get(0).instanceId();
        ec2.waiter().waitUntilInstanceRunning(r ->
r.instanceIds(instanceIdVal));
        System.out.println("Successfully started EC2 instance " +
instanceIdVal + " based on AMI " + amiId);
        return instanceIdVal;

    } catch (SsmException e) {
        System.err.println(e.getMessage());
        System.exit(1);
    }
    return "";
}

// Get a list of instance types.
public static String getInstanceTypes(Ec2Client ec2) {
    String instanceType;
    try {
        DescribeInstanceTypesRequest typesRequest =
DescribeInstanceTypesRequest.builder()
            .maxResults(10)
            .build();

        DescribeInstanceTypesResponse response =
ec2.describeInstanceTypes(typesRequest);
        List<InstanceTypeInfo> instanceTypes = response.instanceTypes();
        for (InstanceTypeInfo type : instanceTypes) {
            System.out.println("The memory information of this type is " +
type.memoryInfo().sizeInMiB());
            System.out.println("Network information is " +
type.networkInfo().toString());
        }
    }
}
```

```
        System.out.println("Instance type is " +
type.instanceType().toString());
        instanceType = type.instanceType().toString();
        if (instanceType.compareTo("t2.2xlarge") == 0){
            return instanceType;
        }
    }

} catch (SsmException e) {
    System.err.println(e.getMessage());
    System.exit(1);
}
return "";
}

// Display the Description field that corresponds to the instance Id value.
public static String describeImage(Ec2Client ec2, String instanceId) {
    try {
        DescribeImagesRequest imagesRequest = DescribeImagesRequest.builder()
            .imageIds(instanceId)
            .build();

        DescribeImagesResponse response = ec2.describeImages(imagesRequest);
        System.out.println("The description of the first image is " +
response.images().get(0).description());
        System.out.println("The name of the first image is " +
response.images().get(0).name());

        // Return the image Id value.
        return response.images().get(0).imageId();

    } catch (SsmException e) {
        System.err.println(e.getMessage());
        System.exit(1);
    }
    return "";
}

// Get the Id value of an instance with amzn2 in the name.
public static String getParaValues(SsmClient ssmClient) {
    try {
        GetParametersByPathRequest parameterRequest =
GetParametersByPathRequest.builder()
            .path("/aws/service/ami-amazon-linux-latest")
```

```
        .build());

        GetParametersByPathIterable responses =
ssmClient.getParametersByPathPaginator(parameterRequest);
        for
(ssoftware.amazon.awssdk.services.ssm.model.GetParametersByPathResponse
response : responses) {
            System.out.println("Test " + response.nextToken());
            List<Parameter> parameterList = response.parameters();
            for (Parameter para : parameterList) {
                System.out.println("The name of the para is: " +
para.name());
                System.out.println("The type of the para is: " +
para.type());
                if (filterName(para.name())) {
                    return para.value();
                }
            }
        }

        } catch (SsmException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
        return "";
    }

    // Return true if the name has amzn2 in it. For example:
    // /aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-arm64-gp2
    private static boolean filterName(String name) {
        String[] parts = name.split("/");
        String myValue = parts[4];
        return myValue.contains("amzn2");
    }

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

            // Use a paginator.
```

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

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

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

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

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

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

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

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

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

public static void describeKeys(Ec2Client ec2) {
    try {
        DescribeKeyPairsResponse response = ec2.describeKeyPairs();
        response.keyPairs().forEach(keyPair -> System.out.printf(
            "Found key pair with name %s " +
                "and fingerprint %s",
            keyPair.keyName(),
            keyPair.keyFingerprint()));
    } catch (Ec2Exception e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
}

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

        CreateKeyPairResponse response = ec2.createKeyPair(request);
        String content = response.keyMaterial();
        BufferedWriter writer = new BufferedWriter(new FileWriter(fileName));
```

```
        writer.write(content);
        writer.close();
        System.out.println("Successfully created key pair named " + keyName);

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

- Para obtener información sobre la API, consulte los siguientes temas en la referencia de la API de AWS SDK for Java 2.x .
  - [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

Hay más información. GitHub Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

Ejecutar un escenario interactivo en un símbolo del sistema.

```
import { mkdtempSync, writeFileSync, rmSync } from "fs";
import { tmpdir } from "os";
import { join } from "path";
import { get } from "http";

import {
  AllocateAddressCommand,
  AssociateAddressCommand,
  AuthorizeSecurityGroupIngressCommand,
  CreateKeyPairCommand,
  CreateSecurityGroupCommand,
  DeleteKeyPairCommand,
  DeleteSecurityGroupCommand,
  DescribeInstancesCommand,
  DescribeKeyPairsCommand,
  DescribeSecurityGroupsCommand,
  DisassociateAddressCommand,
  EC2Client,
  paginateDescribeImages,
  paginateDescribeInstanceTypes,
  ReleaseAddressCommand,
  RunInstancesCommand,
  StartInstancesCommand,
  StopInstancesCommand,
  TerminateInstancesCommand,
  waitUntilInstanceStatusOk,
  waitUntilInstanceStopped,
  waitUntilInstanceTerminated,
} from "@aws-sdk/client-ec2";
import { paginateGetParametersByPath, SSMClient } from "@aws-sdk/client-ssm";
```

```
import { wrapText } from "@aws-doc-sdk-examples/lib/utils/util-string.js";
import { Prompter } from "@aws-doc-sdk-examples/lib/prompter.js";

const ec2Client = new EC2Client();
const ssmClient = new SSMClient();

const prompter = new Prompter();
const confirmMessage = "Continue?";
const tmpDirectory = mkdtempSync(join(tmpdir(), "ec2-scenario-tmp"));

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

  // Save the private key in a temporary location.
  writeFileSync(`${tmpDirectory}/${keyPairName}.pem`, KeyMaterial, {
    mode: 0o400,
  });

  return KeyPairId;
};

const describeKeyPair = async (keyPairName) => {
  const command = new DescribeKeyPairsCommand({
    KeyNames: [keyPairName],
  });
  const { KeyPairs } = await ec2Client.send(command);
  return KeyPairs[0];
};

const createSecurityGroup = async (securityGroupName) => {
  const command = new CreateSecurityGroupCommand({
    GroupName: securityGroupName,
    Description: "A security group for the Amazon EC2 example.",
  });
  const { GroupId } = await ec2Client.send(command);
  return GroupId;
};

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

```

```
const { PublicIp, AllocationId } = await ec2Client.send(command);
return { PublicIp, AllocationId };
};

const getLocalIpAddress = () => {
  return new Promise((res, rej) => {
    get("http://checkip.amazonaws.com", (response) => {
      let data = "";
      response.on("data", (chunk) => (data += chunk));
      response.on("end", () => res(data.trim()));
    }).on("error", (err) => {
      rej(err);
    });
  });
};

const authorizeSecurityGroupIngress = async (securityGroupId) => {
  const ipAddress = await getLocalIpAddress();
  const command = new AuthorizeSecurityGroupIngressCommand({
    GroupId: securityGroupId,
    IpPermissions: [
      {
        IpProtocol: "tcp",
        FromPort: 22,
        ToPort: 22,
        IpRanges: [{ CidrIp: `${ipAddress}/32` }],
      },
    ],
  });
  await ec2Client.send(command);
  return ipAddress;
};

const describeSecurityGroup = async (securityGroupName) => {
  const command = new DescribeSecurityGroupsCommand({
    GroupNames: [securityGroupName],
  });
  const { SecurityGroups } = await ec2Client.send(command);

  return SecurityGroups[0];
};

const getAmznLinux2AMIs = async () => {
```

```
const AMIs = [];
for await (const page of paginateGetParametersByPath(
  {
    client: ssmClient,
  },
  { Path: "/aws/service/ami-amazon-linux-latest" },
)) {
  page.Parameters.forEach((param) => {
    if (param.Name.includes("amzn2")) {
      AMIs.push(param.Value);
    }
  });
}

const imageDetails = [];

for await (const page of paginateDescribeImages(
  { client: ec2Client },
  { ImageIds: AMIs },
)) {
  imageDetails.push(...(page.Images || []));
}

const choices = imageDetails.map((image, index) => ({
  name: `${image.ImageId} - ${image.Description}`,
  value: index,
}));

/**
 * @type {number}
 */
const selectedIndex = await prompter.select({
  message: "Select an image.",
  choices,
});

return imageDetails[selectedIndex];
};

/**
 * @param {import('@aws-sdk/client-ec2').Image} imageDetails
 */
const getCompatibleInstanceTypes = async (imageDetails) => {
  const paginator = paginateDescribeInstanceTypes(
```

```
{ client: ec2Client, pageSize: 25 },
{
  Filters: [
    {
      Name: "processor-info.supported-architecture",
      Values: [imageDetails.Architecture],
    },
    { Name: "instance-type", Values: ["*.micro", "*.small"] },
  ],
},
);

const instanceTypes = [];

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

const choices = instanceTypes.map((type, index) => ({
  name: `${type.InstanceType} - Memory:${type.MemoryInfo.SizeInMiB}`,
  value: index,
})));

/**
 * @type {number}
 */
const selectedIndex = await prompter.select({
  message: "Select an instance type.",
  choices,
});
return instanceTypes[selectedIndex];
};

const runInstance = async ({
  keyPairName,
  securityGroupId,
  imageId,
  instanceType,
}) => {
  const command = new RunInstancesCommand({
    KeyName: keyPairName,
    SecurityGroupIds: [securityGroupId],
```

```
    ImageId: imageId,
    InstanceType: instanceType,
    MinCount: 1,
    MaxCount: 1,
  });

  const { Instances } = await ec2Client.send(command);
  await waitUntilInstanceStatusOk(
    { client: ec2Client },
    { InstanceIds: [Instances[0].InstanceId] },
  );
  return Instances[0].InstanceId;
};

const describeInstance = async (instanceId) => {
  const command = new DescribeInstancesCommand({
    InstanceIds: [instanceId],
  });

  const { Reservations } = await ec2Client.send(command);
  return Reservations[0].Instances[0];
};

const displaySSHConnectionInfo = ({ publicIp, keyPairName }) => {
  return `ssh -i ${tmpDirectory}/${keyPairName}.pem ec2-user@${publicIp}`;
};

const stopInstance = async (instanceId) => {
  const command = new StopInstancesCommand({ InstanceIds: [instanceId] });
  await ec2Client.send(command);
  await waitUntilInstanceStopped(
    { client: ec2Client },
    { InstanceIds: [instanceId] },
  );
};

const startInstance = async (instanceId) => {
  const startCommand = new StartInstancesCommand({ InstanceIds: [instanceId] });
  await ec2Client.send(startCommand);
  await waitUntilInstanceStatusOk(
    { client: ec2Client },
    { InstanceIds: [instanceId] },
  );
  return await describeInstance(instanceId);
};
```

```
};

const associateAddress = async ({ allocationId, instanceId }) => {
  const command = new AssociateAddressCommand({
    AllocationId: allocationId,
    InstanceId: instanceId,
  });

  const { AssociationId } = await ec2Client.send(command);
  return AssociationId;
};

const disassociateAddress = async (associationId) => {
  const command = new DisassociateAddressCommand({
    AssociationId: associationId,
  });
  try {
    await ec2Client.send(command);
  } catch (err) {
    console.warn(
      `Failed to disassociated address with association id: ${associationId}`,
      err,
    );
  }
};

const releaseAddress = async (allocationId) => {
  const command = new ReleaseAddressCommand({
    AllocationId: allocationId,
  });

  try {
    await ec2Client.send(command);
    console.log(`Address with allocation ID ${allocationId} released.\n`);
  } catch (err) {
    console.log(
      `Failed to release address with allocation id: ${allocationId}.`,
      err,
    );
  }
};

const restartInstance = async (instanceId) => {
  console.log("Stopping instance.");
```

```
    await stopInstance(instanceId);
    console.log("Instance stopped.");
    console.log("Starting instance.");
    const { PublicIpAddress } = await startInstance(instanceId);
    return PublicIpAddress;
  };

const terminateInstance = async (instanceId) => {
  const command = new TerminateInstancesCommand({
    InstanceIds: [instanceId],
  });

  try {
    await ec2Client.send(command);
    await waitUntilInstanceTerminated(
      { client: ec2Client },
      { InstanceIds: [instanceId] },
    );
    console.log(`Instance with ID ${instanceId} terminated.\n`);
  } catch (err) {
    console.warn(`Failed to terminate instance ${instanceId}.`, err);
  }
};

const deleteSecurityGroup = async (securityGroupId) => {
  const command = new DeleteSecurityGroupCommand({
    GroupId: securityGroupId,
  });

  try {
    await ec2Client.send(command);
    console.log(`Security group ${securityGroupId} deleted.\n`);
  } catch (err) {
    console.warn(`Failed to delete security group ${securityGroupId}.`, err);
  }
};

const deleteKeyPair = async (keyPairName) => {
  const command = new DeleteKeyPairCommand({
    KeyName: keyPairName,
  });

  try {
    await ec2Client.send(command);
```



```
    console.log(`Key pair ${keyPairName} deleted.\n`);
  } catch (err) {
    console.warn(`Failed to delete key pair ${keyPairName}.`, err);
  }
};

const deleteTemporaryDirectory = () => {
  try {
    rmSync(tmpDirectory, { recursive: true });
    console.log(`Temporary directory ${tmpDirectory} deleted.\n`);
  } catch (err) {
    console.warn(`Failed to delete temporary directory ${tmpDirectory}.`, err);
  }
};

export const main = async () => {
  const keyPairName = "ec2-scenario-key-pair";
  const securityGroupName = "ec2-scenario-security-group";

  let securityGroupId, ipAllocationId, publicIp, instanceId, associationId;

  console.log(wrapText("Welcome to the Amazon EC2 basic usage scenario."));

  try {
    // Prerequisites
    console.log(
      "Before you launch an instance, you'll need a few things:",
      "\n - A Key Pair",
      "\n - A Security Group",
      "\n - An IP Address",
      "\n - An AMI",
      "\n - A compatible instance type",
      "\n\n I'll go ahead and take care of the first three, but I'll need your
      help for the rest.",
    );

    await prompter.confirm({ message: confirmMessage });

    await createKeyPair(keyPairName);
    securityGroupId = await createSecurityGroup(securityGroupName);
    const { PublicIp, AllocationId } = await allocateIpAddress();
    ipAllocationId = AllocationId;
    publicIp = PublicIp;
    const ipAddress = await authorizeSecurityGroupIngress(securityGroupId);
```

```
const { KeyName } = await describeKeyPair(keyPairName);
const { GroupName } = await describeSecurityGroup(securityGroupName);
console.log(`# created the key pair ${KeyName}.\n`);
console.log(
  `# created the security group ${GroupName}`,
  `and allowed SSH access from ${ipAddress} (your IP).\n`,
);
console.log(`# allocated ${publicIp} to be used for your EC2 instance.\n`);

await prompter.confirm({ message: confirmMessage });

// Creating the instance
console.log(wrapText("Create the instance."));
console.log(
  "You get to choose which image you want. Select an amazon-linux-2 image
from the following:",
);
const imageDetails = await getAmznLinux2AMIs();
const instanceTypeDetails = await getCompatibleInstanceTypes(imageDetails);
console.log("Creating your instance. This can take a few seconds.");
instanceId = await runInstance({
  keyPairName,
  securityGroupId,
  imageId: imageDetails.ImageId,
  instanceType: instanceTypeDetails.InstanceType,
});
const instanceDetails = await describeInstance(instanceId);
console.log(`# instance ${instanceId}.\n`);
console.log(instanceDetails);
console.log(
  `
\nYou should now be able to SSH into your instance from another
terminal:`,
  `
\n${displaySSHConnectionInfo({
  publicIp: instanceDetails.PublicIpAddress,
  keyPairName,
})}`,
);

await prompter.confirm({ message: confirmMessage });

// Understanding the IP address.
console.log(wrapText("Understanding the IP address."));
console.log(
```

```
    "When you stop and start an instance, the IP address will change. I'll
restart your",
    "instance for you. Notice how the IP address changes.",
  );
  const ipAddressAfterRestart = await restartInstance(instanceId);
  console.log(
    `
Instance started. The IP address changed from
${instanceDetails.PublicIpAddress} to ${ipAddressAfterRestart}`,
    `
${displaySSHConnectionInfo({
  publicIp: ipAddressAfterRestart,
  keyPairName,
})}`,
  );
  await prompter.confirm({ message: confirmMessage });
  console.log(
    `If you want to the IP address to be static, you can associate an
allocated`,
    `IP address to your instance. I allocated ${publicIp} for you earlier, and
now I'll associate it to your instance.`,
  );
  associationId = await associateAddress({
    allocationId: ipAllocationId,
    instanceId,
  });
  console.log(
    "Done. Now you should be able to SSH using the new IP.\n",
    `${displaySSHConnectionInfo({ publicIp, keyPairName })}`,
  );
  await prompter.confirm({ message: confirmMessage });
  console.log(
    "I'll restart the server again so you can see the IP address remains the
same.",
  );
  const ipAddressAfterAssociated = await restartInstance(instanceId);
  console.log(
    `Done. Here's your SSH info. Notice the IP address hasn't changed.`,
    `
${displaySSHConnectionInfo({
  publicIp: ipAddressAfterAssociated,
  keyPairName,
})}`,
  );
  await prompter.confirm({ message: confirmMessage });
} catch (err) {
  console.error(err);
}
```

```
} finally {
  // Clean up.
  console.log(wrapText("Clean up."));
  console.log("Now I'll clean up all of the stuff I created.");
  await prompter.confirm({ message: confirmMessage });
  console.log("Cleaning up. Some of these steps can take a bit of time.");
  await disassociateAddress(associationId);
  await terminateInstance(instanceId);
  await releaseAddress(ipAllocationId);
  await deleteSecurityGroup(securityGroupId);
  deleteTemporaryDirectory();
  await deleteKeyPair(keyPairName);
  console.log(
    "Done cleaning up. Thanks for staying until the end!",
    "If you have any feedback please use the feedback button in the docs",
    "or create an issue on GitHub.",
  );
}
};
```

- Para obtener información sobre la API, consulte los siguientes temas en la referencia de la API de AWS SDK for JavaScript .
  - [AllocateAddress](#)
  - [AssociateAddress](#)
  - [AuthorizeSecurityGroupIngress](#)
  - [CreateKeyPair](#)
  - [CreateSecurityGroup](#)
  - [DeleteKeyPair](#)
  - [DeleteSecurityGroup](#)
  - [DescribeImages](#)
  - [DescribeInstanceTypes](#)
  - [DescribeInstances](#)
  - [DescribeKeyPairs](#)
  - [DescribeSecurityGroups](#)
  - [DisassociateAddress](#)
  - [ReleaseAddress](#)

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

## Kotlin

### SDK para Kotlin

#### Note

Hay más información GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de 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.waitForInstanceRunning {
        // 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.waitForInstanceStopped {
            // 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 obtener información acerca de la API, consulte los siguientes temas en la Referencia de la API de AWS SDK para Kotlin.
  - [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

Hay más información GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el [Repositorio de ejemplos de código de AWS](#).

## Ejecutar un escenario interactivo en un símbolo del sistema.

```

class Ec2InstanceScenario:
    """Runs an interactive scenario that shows how to get started using EC2
    instances."""

    def __init__(self, inst_wrapper, key_wrapper, sg_wrapper, eip_wrapper,
                 ssm_client):
        """
        :param inst_wrapper: An object that wraps instance actions.
        :param key_wrapper: An object that wraps key pair actions.
        :param sg_wrapper: An object that wraps security group actions.
        :param eip_wrapper: An object that wraps Elastic IP actions.
        :param ssm_client: A Boto3 AWS Systems Manager client.
        """
        self.inst_wrapper = inst_wrapper
        self.key_wrapper = key_wrapper
        self.sg_wrapper = sg_wrapper
        self.eip_wrapper = eip_wrapper
        self.ssm_client = ssm_client

    @demo_func
    def create_and_list_key_pairs(self):
        """
        1. Creates an RSA key pair and saves its private key data as a .pem file
        in secure temporary storage. The private key data is deleted after the example
        completes.
        2. Lists the first five key pairs for the current account.
        """
        print(
            "Let's create an RSA key pair that you can be use to securely connect
            to "
            "your EC2 instance."
        )
        key_name = q.ask("Enter a unique name for your key: ", q.non_empty)
        self.key_wrapper.create(key_name)
        print(
            f"Created a key pair {self.key_wrapper.key_pair.key_name} and saved
            the "
            f"private key to {self.key_wrapper.key_file_path}.\n"
        )
        if q.ask("Do you want to list some of your key pairs? (y/n) ",
                q.is_yesno):

```

```

        self.key_wrapper.list(5)

@demo_func
def create_security_group(self):
    """
    1. Creates a security group for the default VPC.
    2. Adds an inbound rule to allow SSH. The SSH rule allows only
        inbound traffic from the current computer's public IPv4 address.
    3. Displays information about the security group.

    This function uses 'http://checkip.amazonaws.com' to get the current
    public IP
    address of the computer that is running the example. This method works in
    most
    cases. However, depending on how your computer connects to the internet,
    you
    might have to manually add your public IP address to the security group
    by using
    the AWS Management Console.
    """
    print("Let's create a security group to manage access to your instance.")
    sg_name = q.ask("Enter a unique name for your security group: ",
q.non_empty)
    security_group = self.sg_wrapper.create(
        sg_name, "Security group for example: get started with instances."
    )
    print(
        f"Created security group {security_group.group_name} in your default
"
        f"VPC {security_group.vpc_id}.\n"
    )

    ip_response = urllib.request.urlopen("http://checkip.amazonaws.com")
    current_ip_address = ip_response.read().decode("utf-8").strip()
    print("Let's add a rule to allow SSH only from your current IP address.")
    print(f"Your public IP address is {current_ip_address}.")
    q.ask("Press Enter to add this rule to your security group.")
    response = self.sg_wrapper.authorize_ingress(current_ip_address)
    if response["Return"]:
        print("Security group rules updated.")
    else:
        print("Couldn't update security group rules.")
    self.sg_wrapper.describe()

```

```

@demo_func
def create_instance(self):
    """
    1. Gets a list of Amazon Linux 2 AMIs from AWS Systems Manager.
    Specifying the
        '/aws/service/ami-amazon-linux-latest' path returns only the latest
    AMIs.
    2. Gets and displays information about the available AMIs and lets you
    select one.
    3. Gets a list of instance types that are compatible with the selected
    AMI and
        lets you select one.
    4. Creates an instance with the previously created key pair and security
    group,
        and the selected AMI and instance type.
    5. Waits for the instance to be running and then displays its
    information.
    """
    ami_paginator = self.ssm_client.get_paginator("get_parameters_by_path")
    ami_options = []
    for page in ami_paginator.paginate(Path="/aws/service/ami-amazon-linux-
latest"):
        ami_options += page["Parameters"]
    amzn2_images = self.inst_wrapper.get_images(
        [opt["Value"] for opt in ami_options if "amzn2" in opt["Name"]]
    )
    print(
        "Let's create an instance from an Amazon Linux 2 AMI. Here are some
options:"
    )
    image_choice = q.choose(
        "Which one do you want to use? ", [opt.description for opt in
amzn2_images]
    )
    print("Great choice!\n")

    print(
        f"Here are some instance types that support the "
        f"{amzn2_images[image_choice].architecture} architecture of the
image:"
    )
    inst_types = self.inst_wrapper.get_instance_types(
        amzn2_images[image_choice].architecture
    )

```

```
    inst_type_choice = q.choose(
        "Which one do you want to use? ", [it["InstanceType"] for it in
inst_types]
    )
    print("Another great choice.\n")

    print("Creating your instance and waiting for it to start...")
    self.inst_wrapper.create(
        amzn2_images[image_choice],
        inst_types[inst_type_choice]["InstanceType"],
        self.key_wrapper.key_pair,
        [self.sg_wrapper.security_group],
    )
    print(f"Your instance is ready:\n")
    self.inst_wrapper.display()

    print("You can use SSH to connect to your instance.")
    print(
        "If the connection attempt times out, you might have to manually
update "
        "the SSH ingress rule for your IP address in the AWS Management
Console."
    )
    self._display_ssh_info()

    def _display_ssh_info(self):
        """
        Displays an SSH connection string that can be used to connect to a
running
instance.
        """
        print("To connect, open another command prompt and run the following
command:")
        if self.eip_wrapper.elastic_ip is None:
            print(
                f"\tssh -i {self.key_wrapper.key_file_path} "
                f"ec2-user@{self.inst_wrapper.instance.public_ip_address}"
            )
        else:
            print(
                f"\tssh -i {self.key_wrapper.key_file_path} "
                f"ec2-user@{self.eip_wrapper.elastic_ip.public_ip}"
            )
        q.ask("Press Enter when you're ready to continue the demo.")
```

```
@demo_func
def associate_elastic_ip(self):
    """
    1. Allocates an Elastic IP address and associates it with the instance.
    2. Displays an SSH connection string that uses the Elastic IP address.
    """
    print(
        "You can allocate an Elastic IP address and associate it with your
instance\n"
        "to keep a consistent IP address even when your instance restarts."
    )
    elastic_ip = self.eip_wrapper.allocate()
    print(f"Allocated static Elastic IP address: {elastic_ip.public_ip}.")
    self.eip_wrapper.associate(self.inst_wrapper.instance)
    print(f"Associated your Elastic IP with your instance.")
    print(
        "You can now use SSH to connect to your instance by using the Elastic
IP."
    )
    self._display_ssh_info()

@demo_func
def stop_and_start_instance(self):
    """
    1. Stops the instance and waits for it to stop.
    2. Starts the instance and waits for it to start.
    3. Displays information about the instance.
    4. Displays an SSH connection string. When an Elastic IP address is
associated
with the instance, the IP address stays consistent when the instance
stops
and starts.
    """
    print("Let's stop and start your instance to see what changes.")
    print("Stopping your instance and waiting until it's stopped...")
    self.inst_wrapper.stop()
    print("Your instance is stopped. Restarting...")
    self.inst_wrapper.start()
    print("Your instance is running.")
    self.inst_wrapper.display()
    if self.eip_wrapper.elastic_ip is None:
        print(
```

```

        "Every time your instance is restarted, its public IP address
changes."
    )
    else:
        print(
            "Because you have associated an Elastic IP with your instance,
you can \n"
            "connect by using a consistent IP address after the instance
restarts."
        )
    self._display_ssh_info()

@demo_func
def cleanup(self):
    """
    1. Disassociate and delete the previously created Elastic IP.
    2. Terminate the previously created instance.
    3. Delete the previously created security group.
    4. Delete the previously created key pair.
    """
    print("Let's clean everything up. This example created these resources:")
    print(f"\tElastic IP: {self.eip_wrapper.elastic_ip.allocation_id}")
    print(f"\tInstance: {self.inst_wrapper.instance.id}")
    print(f"\tSecurity group: {self.sg_wrapper.security_group.id}")
    print(f"\tKey pair: {self.key_wrapper.key_pair.name}")
    if q.ask("Ready to delete these resources? (y/n) ", q.is_yesno):
        self.eip_wrapper.disassociate()
        print("Disassociated the Elastic IP from the instance.")
        self.eip_wrapper.release()
        print("Released the Elastic IP.")
        print("Terminating the instance and waiting for it to terminate...")
        self.inst_wrapper.terminate()
        print("Instance terminated.")
        self.sg_wrapper.delete()
        print("Deleted security group.")
        self.key_wrapper.delete()
        print("Deleted key pair.")

def run_scenario(self):
    logging.basicConfig(level=logging.INFO, format="%(levelname)s:
%(message)s")

    print("-" * 88)
    print(

```



```

        "Welcome to the Amazon Elastic Compute Cloud (Amazon EC2) get started
with instances demo."
    )
    print("-" * 88)

    self.create_and_list_key_pairs()
    self.create_security_group()
    self.create_instance()
    self.stop_and_start_instance()
    self.associate_elastic_ip()
    self.stop_and_start_instance()
    self.cleanup()

    print("\nThanks for watching!")
    print("-" * 88)

if __name__ == "__main__":
    try:
        scenario = Ec2InstanceScenario(
            InstanceWrapper.from_resource(),
            KeyPairWrapper.from_resource(),
            SecurityGroupWrapper.from_resource(),
            ElasticIpWrapper.from_resource(),
            boto3.client("ssm"),
        )
        scenario.run_scenario()
    except Exception:
        logging.exception("Something went wrong with the demo.")

```

Defina una clase que contenga las acciones del par de claves.

```

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

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

```

```

        :param key_file_dir: The folder where the private key information is
        stored.

            This should be a secure folder.
        :param key_pair: A Boto3 KeyPair object. This is a high-level object that
            wraps key pair actions.
        """
        self.ec2_resource = ec2_resource
        self.key_pair = key_pair
        self.key_file_path = None
        self.key_file_dir = key_file_dir

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

    def create(self, key_name):
        """
        Creates a key pair that can be used to securely connect to an EC2
        instance.

        The returned key pair contains private key information that cannot be
        retrieved
        again. The private key data is stored as a .pem file.

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

```

```
        else:
            return self.key_pair

def list(self, limit):
    """
    Displays a list of key pairs for the current account.

    :param limit: The maximum number of key pairs to list.
    """
    try:
        for kp in self.ec2_resource.key_pairs.limit(limit):
            print(f"Found {kp.key_type} key {kp.name} with fingerprint:")
            print(f"\t{kp.key_fingerprint}")
    except ClientError as err:
        logger.error(
            "Couldn't list key pairs. Here's why: %s: %s",
            err.response["Error"]["Code"],
            err.response["Error"]["Message"],
        )
        raise

def delete(self):
    """
    Deletes a key pair.
    """
    if self.key_pair is None:
        logger.info("No key pair to delete.")
        return

    key_name = self.key_pair.name
    try:
        self.key_pair.delete()
        self.key_pair = None
    except ClientError as err:
        logger.error(
            "Couldn't delete key %s. Here's why: %s : %s",
            key_name,
            err.response["Error"]["Code"],
            err.response["Error"]["Message"],
        )
        raise
```

Defina una clase que ajuste las acciones del grupo de seguridad.

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

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

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

    def create(self, group_name, group_description):
        """
        Creates a security group in the default virtual private cloud (VPC) of
        the
        current account.

        :param group_name: The name of the security group to create.
        :param group_description: The description of the security group to
        create.
        :return: A Boto3 SecurityGroup object that represents the newly created
        security group.
        """
        try:
            self.security_group = self.ec2_resource.create_security_group(
```

```
        GroupName=group_name, Description=group_description
    )
except ClientError as err:
    logger.error(
        "Couldn't create security group %s. Here's why: %s: %s",
        group_name,
        err.response["Error"]["Code"],
        err.response["Error"]["Message"],
    )
    raise
else:
    return self.security_group

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

    :param ssh_ingress_ip: The IP address that is granted inbound access to
connect
                           to port 22 over TCP, used for SSH.
    :return: The response to the authorization request. The 'Return' field of
the
            response indicates whether the request succeeded or failed.
    """
    if self.security_group is None:
        logger.info("No security group to update.")
        return

    try:
        ip_permissions = [
            {
                # SSH ingress open to only the specified IP address.
                "IpProtocol": "tcp",
                "FromPort": 22,
                "ToPort": 22,
                "IpRanges": [{"CidrIp": f"{ssh_ingress_ip}/32"}],
            }
        ]
        response = self.security_group.authorize_ingress(
            IpPermissions=ip_permissions
        )
    except ClientError as err:
        logger.error(
```

```
        "Couldn't authorize inbound rules for %s. Here's why: %s: %s",
        self.security_group.id,
        err.response["Error"]["Code"],
        err.response["Error"]["Message"],
    )
    raise
else:
    return response

def describe(self):
    """
    Displays information about the security group.
    """
    if self.security_group is None:
        logger.info("No security group to describe.")
        return

    try:
        print(f"Security group: {self.security_group.group_name}")
        print(f"\tID: {self.security_group.id}")
        print(f"\tVPC: {self.security_group.vpc_id}")
        if self.security_group.ip_permissions:
            print(f"Inbound permissions:")
            pp(self.security_group.ip_permissions)
    except ClientError as err:
        logger.error(
            "Couldn't get data for security group %s. Here's why: %s: %s",
            self.security_group.id,
            err.response["Error"]["Code"],
            err.response["Error"]["Message"],
        )
        raise

def delete(self):
    """
    Deletes the security group.
    """
    if self.security_group is None:
        logger.info("No security group to delete.")
        return

    group_id = self.security_group.id
```

```

try:
    self.security_group.delete()
except ClientError as err:
    logger.error(
        "Couldn't delete security group %s. Here's why: %s: %s",
        group_id,
        err.response["Error"]["Code"],
        err.response["Error"]["Message"],
    )
    raise

```

Defina una clase que ajuste las acciones de la instancia.

```

class InstanceWrapper:
    """Encapsulates Amazon Elastic Compute Cloud (Amazon EC2) instance
    actions."""

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

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

    def create(self, image, instance_type, key_pair, security_groups=None):
        """
        Creates a new EC2 instance. The instance starts immediately after

```

```

        it is created.

        The instance is created in the default VPC of the current account.

        :param image: A Boto3 Image object that represents an Amazon Machine
Image (AMI)
                that defines attributes of the instance that is created.
The AMI
                defines things like the kind of operating system and the
type of
                storage used by the instance.
        :param instance_type: The type of instance to create, such as 't2.micro'.
                The instance type defines things like the number of
CPUs and
                the amount of memory.
        :param key_pair: A Boto3 KeyPair or KeyPairInfo object that represents
the key
                pair that is used to secure connections to the instance.
        :param security_groups: A list of Boto3 SecurityGroup objects that
represents the
                security groups that are used to grant access to
the
                instance. When no security groups are specified,
the
                default security group of the VPC is used.
        :return: A Boto3 Instance object that represents the newly created
instance.
        """
        try:
            instance_params = {
                "ImageId": image.id,
                "InstanceType": instance_type,
                "KeyName": key_pair.name,
            }
            if security_groups is not None:
                instance_params["SecurityGroupIds"] = [sg.id for sg in
security_groups]
            self.instance = self.ec2_resource.create_instances(
                **instance_params, MinCount=1, MaxCount=1
            )[0]
            self.instance.wait_until_running()
        except ClientError as err:
            logging.error(

```



```
        "Couldn't create instance with image %s, instance type %s, and
key %s. "
        "Here's why: %s: %s",
        image.id,
        instance_type,
        key_pair.name,
        err.response["Error"]["Code"],
        err.response["Error"]["Message"],
    )
    raise
else:
    return self.instance

def display(self, indent=1):
    """
    Displays information about an instance.

    :param indent: The visual indent to apply to the output.
    """
    if self.instance is None:
        logger.info("No instance to display.")
        return

    try:
        self.instance.load()
        ind = "\t" * indent
        print(f"{ind}ID: {self.instance.id}")
        print(f"{ind}Image ID: {self.instance.image_id}")
        print(f"{ind}Instance type: {self.instance.instance_type}")
        print(f"{ind}Key name: {self.instance.key_name}")
        print(f"{ind}VPC ID: {self.instance.vpc_id}")
        print(f"{ind}Public IP: {self.instance.public_ip_address}")
        print(f"{ind}State: {self.instance.state['Name']}")
    except ClientError as err:
        logger.error(
            "Couldn't display your instance. Here's why: %s: %s",
            err.response["Error"]["Code"],
            err.response["Error"]["Message"],
        )
        raise

def terminate(self):
```

```
"""
Terminates an instance and waits for it to be in a terminated state.
"""
if self.instance is None:
    logger.info("No instance to terminate.")
    return

instance_id = self.instance.id
try:
    self.instance.terminate()
    self.instance.wait_until_terminated()
    self.instance = None
except ClientError as err:
    logging.error(
        "Couldn't terminate instance %s. Here's why: %s: %s",
        instance_id,
        err.response["Error"]["Code"],
        err.response["Error"]["Message"],
    )
    raise

def start(self):
    """
    Starts an instance and waits for it to be in a running state.

    :return: The response to the start request.
    """
    if self.instance is None:
        logger.info("No instance to start.")
        return

    try:
        response = self.instance.start()
        self.instance.wait_until_running()
    except ClientError as err:
        logging.error(
            "Couldn't start instance %s. Here's why: %s: %s",
            self.instance.id,
            err.response["Error"]["Code"],
            err.response["Error"]["Message"],
        )
        raise
    else:
```

```
        return response

def stop(self):
    """
    Stops an instance and waits for it to be in a stopped state.

    :return: The response to the stop request.
    """
    if self.instance is None:
        logger.info("No instance to stop.")
        return

    try:
        response = self.instance.stop()
        self.instance.wait_until_stopped()
    except ClientError as err:
        logger.error(
            "Couldn't stop instance %s. Here's why: %s: %s",
            self.instance.id,
            err.response["Error"]["Code"],
            err.response["Error"]["Message"],
        )
        raise
    else:
        return response

def get_images(self, image_ids):
    """
    Gets information about Amazon Machine Images (AMIs) from a list of AMI
    IDs.

    :param image_ids: The list of AMIs to look up.
    :return: A list of Boto3 Image objects that represent the requested AMIs.
    """
    try:
        images = list(self.ec2_resource.images.filter(ImageIds=image_ids))
    except ClientError as err:
        logger.error(
            "Couldn't get images. Here's why: %s: %s",
            err.response["Error"]["Code"],
            err.response["Error"]["Message"],
        )
    )
```

```
        raise
    else:
        return images

def get_instance_types(self, architecture):
    """
    Gets instance types that support the specified architecture and are
    designated
    as either 'micro' or 'small'. When an instance is created, the instance
    type
    you specify must support the architecture of the AMI you use.

    :param architecture: The kind of architecture the instance types must
    support,
                        such as 'x86_64'.
    :return: A list of instance types that support the specified architecture
    and are either 'micro' or 'small'.
    """
    try:
        inst_types = []
        it_paginator = self.ec2_resource.meta.client.get_paginator(
            "describe_instance_types"
        )
        for page in it_paginator.paginate(
            Filters=[
                {
                    "Name": "processor-info.supported-architecture",
                    "Values": [architecture],
                },
                {"Name": "instance-type", "Values": ["*.micro", "*.small"]},
            ]
        ):
            inst_types += page["InstanceTypes"]
    except ClientError as err:
        logger.error(
            "Couldn't get instance types. Here's why: %s: %s",
            err.response["Error"]["Code"],
            err.response["Error"]["Message"],
        )
        raise
    else:
        return inst_types
```

Defina una clase que ajuste las acciones de la IP elástica.

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

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

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

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

        :return: The newly created Elastic IP object. By default, the address is
        not
                               associated with any instance.
        """
        try:
            response =
self.ec2_resource.meta.client.allocate_address(Domain="vpc")
```

```

        self.elastic_ip =
self.ec2_resource.VpcAddress(response["AllocationId"])
    except ClientError as err:
        logger.error(
            "Couldn't allocate Elastic IP. Here's why: %s: %s",
            err.response["Error"]["Code"],
            err.response["Error"]["Message"],
        )
        raise
    else:
        return self.elastic_ip

def associate(self, instance):
    """
    Associates an Elastic IP address with an instance. When this association
    is
    created, the Elastic IP's public IP address is immediately used as the
    public
    IP address of the associated instance.

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

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

```

```
def disassociate(self):
    """
    Removes an association between an Elastic IP address and an instance.
    When the
    association is removed, the instance is assigned a new public IP address.
    """
    if self.elastic_ip is None:
        logger.info("No Elastic IP to disassociate.")
        return

    try:
        self.elastic_ip.association.delete()
    except ClientError as err:
        logger.error(
            "Couldn't disassociate Elastic IP %s from its instance. Here's
            why: %s: %s",
            self.elastic_ip.allocation_id,
            err.response["Error"]["Code"],
            err.response["Error"]["Message"],
        )
        raise

def release(self):
    """
    Releases an Elastic IP address. After the Elastic IP address is released,
    it can no longer be used.
    """
    if self.elastic_ip is None:
        logger.info("No Elastic IP to release.")
        return

    try:
        self.elastic_ip.release()
    except ClientError as err:
        logger.error(
            "Couldn't release Elastic IP address %s. Here's why: %s: %s",
            self.elastic_ip.allocation_id,
            err.response["Error"]["Code"],
            err.response["Error"]["Message"],
        )
        raise
```

- Para obtener información sobre la API, consulte los siguientes temas en la Referencia de la API de AWS SDK para Python (Boto3).
  - [AllocateAddress](#)
  - [AssociateAddress](#)
  - [AuthorizeSecurityGroupIngress](#)
  - [CreateKeyPair](#)
  - [CreateSecurityGroup](#)
  - [DeleteKeyPair](#)
  - [DeleteSecurityGroup](#)
  - [DescribeImages](#)
  - [DescribeInstanceTypes](#)
  - [DescribeInstances](#)
  - [DescribeKeyPairs](#)
  - [DescribeSecurityGroups](#)
  - [DisassociateAddress](#)
  - [ReleaseAddress](#)
  - [RunInstances](#)
  - [StartInstances](#)
  - [StopInstances](#)
  - [TerminateInstances](#)
  - [UnmonitorInstances](#)

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte [Cree recursos de Amazon EC2 mediante un SDK AWS](#). En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.



# Supervise las solicitudes de API de Amazon EC2 mediante Amazon CloudWatch

Puede supervisar las solicitudes de API de Amazon EC2 mediante Amazon CloudWatch, que recopila datos sin procesar y los procesa para convertirlos en métricas legibles prácticamente en tiempo real. Estas métricas proporcionan una forma sencilla de realizar un seguimiento del uso y los resultados de las operaciones de la API de Amazon EC2 a lo largo del tiempo. Esta información le brinda una mejor perspectiva del rendimiento de sus aplicaciones web y le permite identificar y diagnosticar una variedad de problemas. También puede configurar alarmas que vigilen determinados umbrales y enviar notificaciones o tomar medidas específicas cuando se alcancen esos umbrales.

Para obtener más información al respecto CloudWatch, consulta la [Guía del CloudWatch usuario de Amazon](#).

## Important

Las métricas de la API de Amazon EC2 son una función opcional. Debe solicitar el acceso a esta función. Para obtener más información, consulte [the section called “Habilitar las métricas de la API de Amazon EC2”](#).

## Contenido

- [Habilitar las métricas de la API de Amazon EC2](#)
- [Métricas y dimensiones de la API de Amazon EC2](#)
- [Retención de datos métricos](#)
- [Supervisar las solicitudes realizadas en su nombre](#)
- [Facturación](#)
- [Trabajando con Amazon CloudWatch](#)

## Habilitar las métricas de la API de Amazon EC2

Utilice el siguiente procedimiento para solicitar el acceso a esta función para su Cuenta de AWS.

Para solicitar acceso a esta función

1. [AWS Support Centro](#) abierto.
2. Elija Crear caso.
3. Elija Cuenta y facturación.
4. Para el servicio, selecciona Información general y Cómo empezar.
5. En Categoría, selecciona Uso AWS y servicios.
6. Elija Siguiente paso: información adicional.
7. En Subject (Asunto), escriba **Request access to Amazon EC2 API metrics**.
8. En Descripción, escriba **Please grant my account access to Amazon EC2 API metrics. Related page: <https://docs.aws.amazon.com/AWSEC2/latest/APIReference/monitor.html>**. Incluye también la región a la que necesitas acceder.
9. Elija Siguiente paso: Resuelva ahora o póngase en contacto con nosotros.
10. En la pestaña Comuníquese con nosotros, elija el idioma de contacto y el método de contacto que prefiera.
11. Seleccione Submit (Enviar).

## Métricas y dimensiones de la API de Amazon EC2

### Métricas

Las métricas de la API de Amazon EC2 se incluyen en el AWS/EC2/API espacio de nombres. En las siguientes tablas se enumeran las métricas disponibles para las solicitudes de API de Amazon EC2.

Métrica	Descripción
<code>ClientErrors</code>	<p>El número de solicitudes de API fallidas causadas por errores del cliente.</p> <p>Por lo general, estos errores se deben a una acción del cliente, como especificar un parámetro incorrecto o no válido en la solicitud o al uso de una acción o un recurso en nombre de un usuario que no tiene permiso para usar la acción o el recurso.</p>

Métrica	Descripción
	Unidad: recuento
RequestLimitExceeded	<p>El número de veces que se ha superado la tasa máxima de solicitudes permitida por las API de Amazon EC2 para su cuenta.</p> <p>Las solicitudes de API de Amazon EC2 se limitan para ayudar a mantener el rendimiento del servicio. Si sus solicitudes se han limitado, aparece el error. <code>Client.RequestLimitExceeded</code></p> <p>Unidad: recuento</p>
ServerError	<p>El número de solicitudes de API fallidas causadas por errores internos del servidor.</p> <p>Por lo general, estos errores se deben a un error, una excepción o una falla del AWS servidor.</p> <p>Unidad: recuento</p>
SuccessfulCalls	<p>El número de solicitudes de API que se han realizado correctamente.</p> <p>Unidad: recuento</p>

## Dimensiones

Los datos de las métricas de Amazon EC2 se pueden filtrar en todas las acciones de la API de EC2. Para obtener más información sobre las dimensiones, consulta [Amazon CloudWatch Concepts](#).

## Retención de datos métricos

Las métricas de la API de Amazon EC2 se envían a CloudWatch intervalos de 1 minuto. CloudWatch conserva los datos de las métricas de la siguiente manera:

- Los puntos de datos con un periodo de 60 segundos (1 minuto) están disponibles durante 15 días.

- Los puntos de datos con un período de 300 segundos (5 minutos) están disponibles durante 63 días.
- Los puntos de datos con un período de 3600 segundos (1 hora) están disponibles durante 455 días (15 meses).

## Supervisar las solicitudes realizadas en su nombre

Las solicitudes de API realizadas por AWS los servicios en tu nombre, como las solicitudes realizadas por roles vinculados a servicios, no se tienen en cuenta para los límites de limitación de la API y no envían métricas a Amazon CloudWatch para tu cuenta. Estas solicitudes no se pueden monitorear mediante. CloudWatch

Las solicitudes de API realizadas en tu nombre por proveedores de servicios externos se tienen en cuenta para los límites de limitación de las API y, además, envían las métricas de tu cuenta a Amazon CloudWatch . Estas solicitudes se pueden monitorizar mediante. CloudWatch

## Facturación

Se aplican CloudWatch precios y cargos estándar. No se aplican cargos adicionales por el uso de las métricas de la API de Amazon EC2. Para obtener más información, consulta los [CloudWatch precios de Amazon](#).

## Trabajando con Amazon CloudWatch

Contenido

- [Visualización de CloudWatch las métricas](#)
- [Creando alarmas CloudWatch](#)

## Visualización de CloudWatch las métricas

Utilice el siguiente procedimiento para ver las métricas de la API de Amazon EC2.

Requisito previo

Debe habilitar el acceso a las métricas de la API de Amazon EC2 para su cuenta. Para obtener más información, consulte [the section called “Habilitar las métricas de la API de Amazon EC2”](#).

Para ver las métricas de la API de Amazon EC2 mediante la consola

1. Abra la CloudWatch consola en <https://console.aws.amazon.com/cloudwatch/>.
2. En el panel de navegación, elija Métricas, Todas las métricas.
3. En la pestaña Examinar, elija el espacio de nombres de métricas de EC2/API.
4. Para ver las métricas, seleccione la dimensión de métricas.

Para ver las métricas de la API de Amazon EC2 mediante la línea de comandos

Utilice uno de los siguientes comandos:

- [list-metrics](#) ()AWS CLI

```
aws cloudwatch list-metrics --namespace "AWS/EC2/API"
```

- [Get-CW MetricList](#) ()AWS Tools for Windows PowerShell

```
Get-CWMetricList -Namespace "AWS/EC2/API"
```

## Creando alarmas CloudWatch

Puede crear una CloudWatch alarma que envíe un mensaje de Amazon SNS cuando la alarma cambie de estado. Una alarma vigila una métrica determinada durante el periodo especificado. Envía una notificación a un tema de SNS en función del valor de la métrica en relación con un umbral determinado durante varios periodos de tiempo.

Por ejemplo, puedes crear una alarma que supervise el número de solicitudes de DescribeInstances API que fallan debido a errores del servidor. La siguiente alarma envía una notificación por correo electrónico cuando el número de errores en las solicitudes de DescribeInstances API alcanza un umbral de 10 errores del lado del servidor durante un período de 5 minutos.

Requisito previo

Debe habilitar el acceso a las métricas de la API de Amazon EC2 de su cuenta. Para obtener más información, consulte [the section called "Habilitar las métricas de la API de Amazon EC2"](#).

## Para crear una alarma por errores en el servidor de solicitudes de DescribeInstances API de Amazon EC2

1. Abra la CloudWatch consola en <https://console.aws.amazon.com/cloudwatch/>.
2. En el panel de navegación, elija Alarms (Alarmas) y, luego, Create Alarm (Crear alarma).
3. Elija Create alarm (Crear alarma).
4. Elija Seleccionar métrica y, a continuación, especifique lo siguiente:
  - a. Elija EC2/API.
  - b. Elija métricas por acción.
  - c. Seleccione la casilla de verificación situada junto a la DescribeInstances que está en la misma fila que el nombre de la ServerErrors métrica.
  - d. Elija Seleccionar métrica.
5. Aparece la página Specify metric and conditions (Especificar métrica y condiciones), que muestra un gráfico y otra información sobre la métrica y la estadística que ha seleccionado.
  - a. En Métrica, especifique lo siguiente:
    - i. En Statistic (Estadística), elija Sum (Suma).
    - ii. En Período, compruebe que esté seleccionada la opción 5 minutos.
  - b. En Conditions (Condiciones), especifique lo siguiente:
    - i. En Threshold type (Tipo de umbral), elija Static (Estático).
    - ii. En Siempre que ServerErrors sea, elija Mayor/Igual >=.
    - iii. Por más de... , introduzca 10.
  - c. Elija Siguiente.
6. La página Configure actions (Configurar acciones) aparecerá.
  - En Notificación, especifique lo siguiente:
    - i. Para activar el estado de alarma, seleccione En alarma.
    - ii. En Seleccione un tema de SNS, elija Seleccionar un tema de SNS existente o Crear un tema nuevo y complete los campos obligatorios de la notificación.
    - iii. Elija Siguiente.
7. Aparece la página Añadir nombre y descripción.

- a. En Nombre de la alarma, introduce un nombre para la alarma. El nombre solo debe contener caracteres ASCII.
  - b. En Descripción de la alarma, introduce una descripción opcional para la alarma.
  - c. Elija Siguiente.
8. Aparece la página de previsualización y creación. Compruebe que la información es correcta y, a continuación, seleccione Crear alarma.

Para obtener más información, consulta [Uso de CloudWatch alarmas de Amazon](#) en la Guía del CloudWatch usuario de Amazon.

Las traducciones son generadas a través de traducción automática. En caso de conflicto entre la traducción y la versión original de inglés, prevalecerá la versión en inglés.