aws

Guía del usuario de

Amazon EC2 Auto Scaling



Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Amazon EC2 Auto Scaling: Guía del usuario de

Copyright © 2026 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

¿Qué es Amazon EC2 Auto Scaling? 1
Características de Amazon EC2 Auto Scaling 1
Precios de Amazon EC2 Auto Scaling 4
Introducción 4
Trabajo con grupos de Auto Scaling 4
Beneficios del escalado automático 5
Ejemplo: cubrir una demanda variable 6
Ejemplo: Arquitectura de aplicaciones web 8
Ejemplo: distribuir instancias entre zonas de disponibilidad 9
Ciclo de vida de la instancia 12
Escalado ascendente 13
Instancias en servicio 14
Reducción horizontal 15
Desconexión de una instancia 16
Asociación de una instancia 16
Enlaces de ciclo de vida 16
Entrada y salida del modo de espera 17
Cuotas de Amazon EC2 Auto Scaling 17
Solicitud de limitación para la API de Amazon EC2 Auto Scaling 19
Tasas de terminación de EC2 19
Otros servicios 20
Configuración 21
Prepárese para usar el AWS CLI 21
Introducción 22
Tutorial: Creación de su primer grupo de escalado automático 23
Prepararse para el tutorial 23
Paso 1: crear una plantilla de inicialización 24
Paso 2: Crear un grupo de Auto Scaling de instancia única 25
Paso 3: Verificar el grupo de Auto Scaling 26
Paso 4: Terminar una instancia en el grupo de Auto Scaling 27
Paso 5: Siguientes pasos 28
Paso 6: limpiar 29
Tutorial: Configuración de una aplicación con escalado y balanceo de carga aplicados 30
Requisitos previos 32

Paso 1: Configurar una plantilla de lanzamiento o una configuración de lanzamiento 33
Paso 2: Crear un grupo de Auto Scaling) 37
Paso 3: Verificar que el balanceador de carga está adjunto 38
Paso 4: Siguientes pasos 39
Paso 5: Eliminar 39
Recursos relacionados 41
Plantillas de inicialización 42
Permisos para trabajar con plantillas de lanzamiento 43
Operaciones de la API compatibles con plantillas de lanzamiento 43
Creación de una plantilla de lanzamiento para un grupo de Auto Scaling 44
Creación de una plantilla de lanzamiento (consola) 44
Cambo de la configuración de la interfaz de red predeterminada (consola) 47
Modificación de la configuración de almacenamiento (consola) 50
Creación de una plantilla de lanzamiento a partir de una instancia existente (consola) 53
Recursos relacionados 54
Limitaciones 54
Creación de una plantilla de lanzamiento mediante la configuración avanzada 54
Configuración necesaria 55
Configuración avanzada 55
Solicitar instancias de subasta 60
Capacity Blocks para ML 62
Migración de sus grupos de escalado automático a plantillas de lanzamiento 67
Paso 1: buscar grupos de escalado automático que utilicen configuraciones de lanzamiento 68
Paso 2: copiar una configuración de lanzamiento en una plantilla de lanzamiento 70
Paso 3: actualizar un grupo de escalado automático para utilizar una plantilla de lanzamiento 72
Paso 4: reemplazar sus instancias 73
Información adicional 73
Migre CloudFormation las pilas para lanzar plantillas 73
Búsqueda de grupos de escalado automático que utilizan una configuración de lanzamiento 74
Actualización de una pila para utilizar una plantilla de lanzamiento 75
Comprensión del comportamiento de actualización de los recursos de la pila 79
Seguimiento de la migración 80
Referencia de mapeo de configuración de lanzamiento 80

AWS CLI ejemplos de cómo trabajar con plantillas de lanzamiento 82
Ejemplo de uso 83
Creación de una plantilla de lanzamiento básica 83
Especificar etiquetas que etiquetan instancias en el lanzamiento 84
Especificar un rol de IAM para transferir a instancias 85
Asignación de direcciones IP públicas 85
Especificar un script de datos de usuario que configura instancias en el lanzamiento 86
Especificar una asignación de dispositivos de bloques 86
Especificar hosts dedicados para traer licencias de software de proveedores externos 86
Especificar una interfaz de red existente 87
Creación de múltiples interfaces de red 87
Administración de las plantillas de lanzamiento 88
Actualización de un grupo de Auto Scaling para utilizar una plantilla de lanzamiento 91
Use parámetros de Systems Manager en lugar de ID de AMI 92
Creación de una plantilla de lanzamiento que especifique un parámetro para la AMI 92
Verificación de que la plantilla de lanzamiento tenga el ID de AMI correcto 97
Recursos relacionados 98
Limitaciones 98
Configuraciones de lanzamiento 100
Crear una configuración de lanzamiento 101
Crear una configuración de lanzamiento 102
Configuración de IMDS 105
Crear una configuración de lanzamiento con una instancia EC2 108
Cambio en una configuración de lanzamiento 113
Grupos de escalado automático 115
Creación de grupos de escalado automático mediante plantillas de lanzamiento 117
Creación de un grupo mediante una plantilla de lanzamiento 117
Creación de un grupo mediante el asistente de lanzamiento de EC2 121
Uso de varios tipos de instancia y opciones de compra 126
Creación de grupos de escalado automático mediante configuraciones de lanzamiento 177
Crear un grupo mediante una configuración de lanzamiento 179
Cree un grupo a partir de una instancia usando AWS CLI 182
Aprovisionamiento sincrónico 188
Aprovisionamiento sincrónico y escalado asíncrono 189
Limitaciones y consideraciones 189
Lanzamiento de instancias con aprovisionamiento sincrónico 191

Actualización de un grupo de escalado automático 194
Actualización de las instancias de escalado automático 196
Estrategia de asignación del grupo de escalado automático y cambios en la capacidad 197
Etiquetar grupos e instancias 197
Restricciones de nombres y uso de las etiquetas 198
Ciclo de vida de etiquetado de las instancias EC2 199
Etiqueta los grupos de Auto Scaling 199
Eliminar etiquetas 203
Etiquetas para seguridad 204
Control del acceso a las etiquetas 205
Uso de etiquetas para filtrar grupos de Auto Scaling 206
Políticas de mantenimiento de instancias 210
Descripción general de 210
Establecimiento de una política de mantenimiento de instancias en su grupo 219
Enlaces de ciclo de vida 224
Disponibilidad de los enlaces de ciclo de vida 225
Consideraciones y limitaciones 226
Recursos relacionados 228
Cómo funcionan los enlaces de ciclo de vida en los grupos de escalado automático 229
Preparación para agregar un enlace de ciclo de vida 232
Controle la retención de instancias con políticas de ciclo de vida de las instancias 242
Recuperar el estado de ciclo de vida de destino 248
Incorporación de los enlaces de ciclo de vida en su grupo de escalado automático 250
Cómo completar una acción del ciclo de vida en un grupo de escalado automático 254
Tutorial: Uso de los metadatos de usuario de la instancia para recuperar el estado de ciclo de vida 256
Tutorial: Configuración de un enlace de ciclo de vida que invoca una función Lambda 265
Grupos de calentamiento 275
Conceptos clave 276
Requisitos previos 279
Actualización de las instancias de un grupo en caliente 280
Recursos relacionados 280
Limitaciones 281
Uso de enlaces de ciclo de vida 282
Creación de un grupo en caliente para un grupo de escalado automático 287
Visualización del estado de la comprobación de estado 289

AWS CLI ejemplos para trabajar con piscinas calientes ... 293
Cambio zonal del grupo Auto Scaling ... 296
Conceptos de cambio zonal del grupo Auto Scaling ... 296
Cómo funciona el cambio zonal para los grupos de Auto Scaling ... 297
Mejores prácticas para usar el cambio zonal ... 299
Habilite el cambio zonal mediante el o el Consola de administración de AWS AWS CLI ... 300
Distribución de zonas de disponibilidad ... 303
Detach-attach instancias ... 305
Consideraciones sobre la desconexión de instancias ... 305
Consideraciones sobre la asociación de instancias ... 306
Mueva una instancia a un grupo diferente mediante la opción de desconectar y asociar ... 307
Eliminación temporal de las instancias ... 312
Cómo funciona el estado en espera ... 313
Consideraciones ... 313
Estado de una instancia cuando está en espera ... 314
Eliminación temporal de una instancia al ponerla en espera ... 313
Eliminación de la infraestructura de Auto Scaling ... 319
Eliminar el grupo de Auto Scaling ... 319
(Opcional) Eliminar la configuración de lanzamiento ... 320
(Opcional) Eliminar la plantilla de lanzamiento ... 321
(Opcional) Eliminar el balanceador de carga y los grupos de destino ... 322
(Opcional) Elimine CloudWatch las alarmas ... 323
Protección contra la eliminación de recursos ... 324
Reemplazo de sus instancias ... 331
Actualización de instancias ... 332
Cómo funciona una actualización de instancias ... 333
Comprensión de los valores predeterminados ... 340
Inicio de una actualización de instancias ... 343
Supervise la actualización de una instancia ... 357
Sustituya los volúmenes raíz ... 360
Cancelación de una actualización de instancias ... 365
Inversión de cambios con una reversión ... 366
Uso de la omisión de coincidencias ... 371
Incorporación de puntos de comprobación ... 381
Duración máxima de la instancia ... 387
Consideraciones ... 388

Configuración de la duración máxima de la instancia ... 388
Limitaciones ... 390
Escalar un grupo ... 391
Elección de su método de escalado ... 392
Establecimiento de límites de escalado ... 393
Establecimiento de la preparación predeterminada de instancias ... 395
Consideraciones sobre el rendimiento de escalado ... 396
Elección del tiempo de preparación predeterminada de instancias ... 397
Habilitación de la preparación predeterminada de instancias para un grupo ... 398
Verificación del tiempo de preparación predeterminada de instancias para un grupo ... 400
Búsqueda de políticas de escalado con un tiempo de preparación de instancias previamente establecido ... 401
Borrado de la preparación de instancias previamente establecida para una política de escalado ... 402
Escalado manual ... 402
Cambio de la capacidad deseada de su grupo de escalado automático ... 403
Finalización de una instancia en su grupo de escalado automático (AWS CLI) ... 407
Escalado programado ... 408
Cómo funciona el escalado programado ... 409
Programas recurrentes ... 409
Zona horaria ... 410
Consideraciones ... 411
Limitaciones ... 412
Creación de una acción programada ... 412
Cómo ver detalles de acciones programadas ... 414
Eliminación de una acción programada ... 415
Escalado dinámico ... 416
Funcionamiento de las políticas de escalado dinámico ... 417
Varias políticas de escalado dinámico ... 418
Políticas de escalado de seguimiento de destino ... 420
Políticas de escalado sencillo y por pasos ... 438
Recuperaciones de escalado ... 456
Política de escalado basada en Amazon SQS ... 460
Verificación de una actividad de escalado ... 468
Desactivación de una política de escalado ... 470
Eliminación de una política de escalado para un grupo de escalado automático ... 473

AWS CLI ejemplos de políticas de escalado ... 475
Escalado predictivo ... 479
Funcionamiento del escalado predictivo ... 480
Creación de una política de escalado predictivo ... 484
Evaluación de las políticas de escalado predictivo ... 493
Anulación del pronóstico ... 502
Uso de métricas personalizadas ... 508
Control de la terminación de instancias ... 520
Escenarios de políticas de terminación ... 520
Configuración de políticas de terminación ... 524
Creación de una política de terminación personalizada con Lambda ... 531
Uso de la protección de reducción horizontal de instancias ... 538
Diseño para una terminación de instancias eficiente ... 543
Suspensión-reanudación de procesos ... 551
Tipos de procesos ... 552
Consideraciones ... 553
Suspensión de procesos ... 554
Reanudación de los procesos ... 555
Efecto de la suspensión de procesos sobre otros procesos ... 556
Supervisión ... 560
Comprobaciones de estado ... 562
Acerca de las comprobaciones de estado ... 564
Establecimiento del período de gracia de la comprobación de estado ... 571
Supervisión de volúmenes dañados de Amazon EBS ... 574
Configuración de una comprobación de estado personalizada ... 578
Visionado del motivo de los errores de una comprobación de estado ... 579
Usa las comprobaciones de estado de las aplicaciones ... 581
Solución de problemas de instancias en mal estado ... 584
Supervise con Panel de estado ... 588
Supervise CloudWatch las métricas ... 589
Visualización de gráficos de supervisión en la consola de Amazon EC2 Auto Scaling ... 590
CloudWatch métricas para Amazon EC2 Auto Scaling ... 595
Configuración de la supervisión para instancias de Auto Scaling ... 604
Registre las llamadas a la API mediante CloudTrail ... 607
Eventos de administración de Auto Scaling en CloudTrail ... 608
Ejemplos de eventos de escalado automático ... 608

Auto RemoveAction Scaling activa CloudWatch 610
Opciones de notificaciones de Amazon SNS 610
Amazon SNS y Amazon EC2 Auto Scaling 610
Trabajar con otros servicios 617
Reequilibrio de la capacidad 617
Descripción general de 618
Comportamiento de reequilibrio de la capacidad 619
Consideraciones 620
Habilitar el reequilibrio de la capacidad 622
Reservas de capacidad 629
Preferencia de reserva de capacidad 629
Equilibrio de la zona de disponibilidad y las reservas de capacidad 631
Utilizar la preferencia de reserva de capacidad 631
Reservas de capacidad interrumpible 633
AWS CloudShell 638
CloudFormation 638
Amazon EC2 Auto CloudFormation Scaling y plantillas 639
Obtenga más información sobre CloudFormation 639
Compute Optimizer 640
Limitaciones 640
Hallazgos 641
Ver recomendaciones 641
Consideraciones para evaluar las recomendaciones 642
Elastic Load Balancing 643
Tipos de Elastic Load Balancing 645
Preparación para asociar un equilibrador de carga 646
Asociar un equilibrador de carga 649
Configuración del equilibrador de carga 653
Verificación del estado de asociación 654
Agregar una zona de disponibilidad 656
Eliminar una zona de disponibilidad 658
Desconexión de un balanceador de carga 652
AWS CLI ejemplos para trabajar con Elastic Load Balancing 660
VPC Lattice 668
Preparación para asociar un grupo de destino 670
Asociación de un grupo de destino de VPC Lattice 673

Verificación del estado de asociación ... 678
EventBridge ... 680
Referencia de evento de Amazon EC2 Auto Scaling ... 680
Ejemplos de eventos y patrones de actualización de instancias ... 687
Ejemplos de eventos y patrones de grupos en caliente ... 695
EventBridge reglas ... 700
Amazon VPC ... 706
VPC predeterminada ... 707
VPC no predeterminada ... 707
Consideraciones a la hora de elegir subredes de VPC ... 707
Direcciones IP en una VPC ... 708
Interfaces de red en una VPC ... 709
Tenencia de ubicación de instancias ... 709
AWS Outposts ... 709
Más recursos para obtener más información sobre VPCs ... 710
Seguridad ... 711
Seguridad de la infraestructura ... 712
Recursos relacionados ... 712
Resiliencia ... 712
Recursos relacionados ... 714
Protección de datos ... 714
Úselo AWS KMS keys para cifrar volúmenes de Amazon EBS ... 715
Recursos relacionados ... 716
AWS KMS política de claves para su uso con volúmenes cifrados ... 716
Gestión de identidad y acceso ... 723
Control de acceso ... 723
Cómo funciona Amazon EC2 Auto Scaling con IAM ... 724
Permisos de la API ... 733
Políticas administradas ... 734
Roles vinculados a servicios ... 740
Ejemplos de políticas basadas en identidades ... 745
Prevención de la sustitución confusa entre servicios ... 755
Control del uso de la plantilla de inicialización de Amazon EC2 en los grupos de escalado automático ... 758
Rol de IAM para aplicaciones que se ejecuten en instancias de Amazon EC2 ... 767
Validación de conformidad ... 770

Conformidad con DSS PCI 770
Uso de puntos de conexión de VPC para conectividad privada 771
Creación de un punto de conexión de la VPC de tipo interfaz 771
Creación de una política de puntos de conexión de VPC 772
Trabajando con AWS SDKs 773
Trabajar con las API 775
Ejemplos de código 779
Conceptos básicos 781
Introducción al escalado automático 783
Conceptos básicos 793
Acciones 895
Escenarios 1129
Creación y administración de un servicio resiliente 1129
Solución de problemas 1298
Recuperación de un mensaje de error 1298
Desactivación de las actividades de escalado 1300
Recursos adicionales de solución de problemas 1301
Error de lanzamiento de instancias 1302
La configuración solicitada no se admite actualmente. 1303
El grupo de seguridad <nombre del grupo de seguridad > no existe. El lanzamiento de la instancia EC2 ha producido un error. 1304
El par de claves <par de claves asociado a la instancia EC2> no existe. El lanzamiento de la instancia EC2 ha producido un error. 1304
el tipo de instancia solicitado (<tipo de instancia>) ya no es compatible con la zona de disponibilidad solicitada (<zona de disponibilidad de la instancia>)... 1305
Su precio de solicitud de spot de 0,015 es menor que el precio mínimo requerido de cumplimiento de solicitud de spot de 0,0735... 1305
Nombre de dispositivo inválido <nombre de dispositivo>/Carga de nombre de dispositivo inválido. El lanzamiento de la instancia EC2 ha producido un error. 1306
El valor (<nombre asociado al dispositivo de almacenamiento de la instancia>) del parámetro virtualName no es válido... El lanzamiento de la instancia EC2 ha producido un error. 1306
Los mapeos de dispositivos de bloques de EBS no son compatibles con instance-store. AMIs 1307
Los grupos de ubicación no se pueden utilizar con instancias de tipo “<tipo de instancia>”. El lanzamiento de la instancia EC2 ha producido un error. 1307

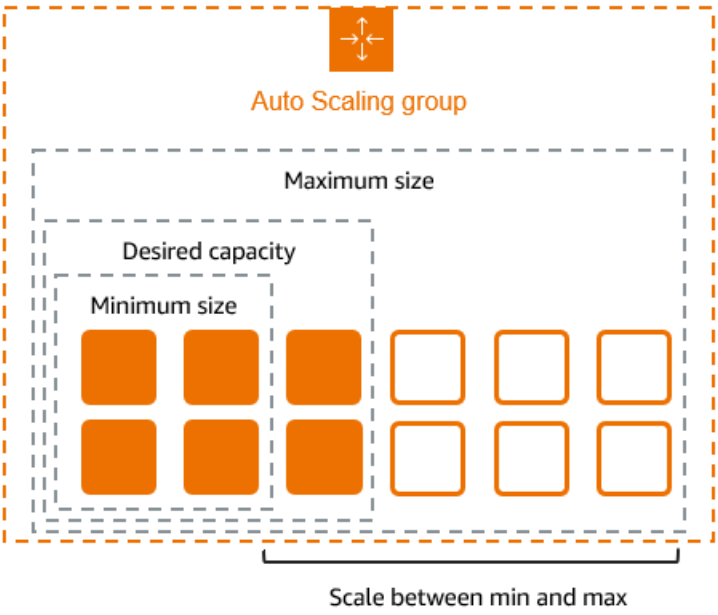
Cliente. InternalError: Error del cliente al iniciarse. 1308
En la actualidad, no dispone de suficiente capacidad de <tipo de instancia> en la zona de disponibilidad que ha solicitado... El lanzamiento de la instancia EC2 ha producido un error. 1309
La reserva solicitada no tiene suficiente capacidad compatible y disponible para esta solicitud. El lanzamiento de la instancia EC2 ha producido un error. 1310
Su reserva de bloques de capacidad <id de reserva>aún no está activa. El lanzamiento de la instancia EC2 ha producido un error. 1310
No hay capacidad de spot disponible que coincida con su solicitud. El lanzamiento de la instancia EC2 ha producido un error. 1311
Ya se están ejecutando <número de instancias> instancias. El lanzamiento de la instancia EC2 ha producido un error. 1311
Problemas con las AMI 1312
El ID de AMI <ID de la AMI> no existe. El lanzamiento de la instancia EC2 ha producido un error. 1313
La AMI <ID de AMI> está pendiente y no se puede ejecutar. El lanzamiento de la instancia EC2 ha producido un error. 1313
Nombre de dispositivo no válido <nombre de dispositivo>. El lanzamiento de la instancia EC2 ha producido un error. 1313
La arquitectura “arm64” del tipo de instancia especificado no coincide con la arquitectura “x86_64” de la AMI especificada... Falló el lanzamiento de la instancia de EC2. 1314
La AMI “<ID de AMI>” está deshabilitada y no se puede ejecutar. El lanzamiento de la instancia EC2 ha producido un error. 1315
Problemas del balanceador de carga 1316
No se encontraron uno o varios grupos de destino. Error al validar la configuración del balanceador de carga. 1317
No se encuentra el equilibrador de carga <su equilibrador de carga>. Error al validar la configuración del balanceador de carga. 1317
No hay ningún balanceador de carga ACTIVO denominado <nombre del balanceador de carga>. Error al actualizar la configuración del balanceador de carga. 1318
La instancia EC2 <ID de instancia> no está en VPC. Error al actualizar la configuración del balanceador de carga. 1318
Problemas de las plantillas de lanzamiento 1318
Debe usar una plantilla de lanzamiento válida y completa (valor no válido) 1318
No cuenta con autorización para utilizar la plantilla de lanzamiento (permisos insuficientes) 1319

Información relacionada 1321
Historial de revisión 1323
..... mccclxxii

¿Qué es Amazon EC2 Auto Scaling?

Amazon EC2 Auto Scaling lo ayuda a garantizar que cuenta con la cantidad correcta de instancias de Amazon EC2 disponibles para gestionar la carga de su aplicación. Crea colecciones de instancias EC2, denominadas grupos de Auto Scaling. Puede especificar el número mínimo de instancias en cada grupo de escalado automático y Amazon EC2 Auto Scaling garantizará que el grupo nunca tenga menos de esas instancias. Puede especificar el número máximo de instancias en cada grupo de escalado automático y Amazon EC2 Auto Scaling garantizará que el grupo nunca tenga más de esas instancias. Si especifica la capacidad deseada, cuando crea el grupo o con posterioridad, Amazon EC2 Auto Scaling garantizará que el grupo tenga ese número de instancias. Si especifica políticas de escalado, Amazon EC2 Auto Scaling puede lanzar o terminar instancias conforme aumente o disminuya la demanda de su aplicación.

Por ejemplo, el siguiente grupo de escalado automático tiene un tamaño mínimo de cuatro instancias, una capacidad deseada de seis instancias y un tamaño máximo de doce instancias. Las políticas de escalado que defina ajustan el número de instancias, en el número mínimo y máximo de instancias, en función de los criterios que especifique.



Características de Amazon EC2 Auto Scaling

Con Amazon EC2 Auto Scaling, las instancias de EC2 se organizan en grupos de escalado automático para que puedan tratarse como una unidad lógica a efectos de escalado y

administración. Los grupos de escalado automático utilizan plantillas de lanzamiento (o configuraciones de lanzamiento) como plantillas de configuración para las instancias de EC2.

Estas son algunas de las principales características de Amazon EC2 Auto Scaling:

Supervisión del estado de las instancias en ejecución

Amazon EC2 Auto Scaling supervisa automáticamente el estado y la disponibilidad de las instancias mediante comprobaciones de estado de EC2 y reemplaza las instancias terminadas o deterioradas para mantener la capacidad deseada.

Comprobaciones de estado personalizadas

Además de las comprobaciones de estado integradas, puede definir comprobaciones de estado personalizadas que sean específicas de su aplicación para comprobar que responda de la forma esperada. Si una instancia no supera la comprobación de estado personalizada, se reemplaza automáticamente para mantener la capacidad deseada.

Equilibrio de capacidad entre zonas de disponibilidad

Puede especificar varias zonas de disponibilidad para el grupo de escalado automático, y Amazon EC2 Auto Scaling equilibrará las instancias de manera uniforme en las zonas de disponibilidad a medida que el grupo escala. Esto proporciona una alta disponibilidad y resiliencia al proteger sus aplicaciones de los fallos en una única ubicación.

Varios tipos de instancia y opciones de compra

Dentro de un único grupo de Auto Scaling, puede lanzar varios tipos de instancias y opciones de compra (puntuales e On-Demand instancias), lo que le permite optimizar los costos mediante el uso de instancias puntuales. También puede aprovechar los descuentos de Reserved Instance y Savings Plans utilizándolos junto con On-Demand las instancias del grupo.

Sustitución automática de instancias de spot

Si su grupo incluye instancias de spot, Amazon EC2 Auto Scaling puede solicitar automáticamente la sustitución de la capacidad de spot si sus instancias de spot se ven interrumpidas. Mediante el reequilibrio de capacidad, Amazon EC2 Auto Scaling también puede supervisar y reemplazar de manera proactiva las instancias de spot que corren un riesgo elevado de interrupción.

Equilibrio de carga

Puede usar la equilibración de carga y las comprobaciones de estado de Elastic Load Balancing para garantizar una distribución uniforme del tráfico de aplicaciones a las instancias en buen

estado. Siempre que se inician o terminan instancias, Amazon EC2 Auto Scaling registra y anula automáticamente el registro de las instancias en el equilibrador de carga.

Escalabilidad

Amazon EC2 Auto Scaling también ofrece varias formas de escalar sus grupo de escalado automático. El uso del escalado automático le permite mantener la disponibilidad de las aplicaciones y reducir los costos al añadir capacidad para gestionar los picos de carga y eliminar la capacidad cuando la demanda es menor. También puede ajustar manualmente el tamaño del grupo de escalado automático según sea necesario.

Actualización de instancias

La característica de actualización de instancias proporciona un mecanismo para actualizar las instancias de forma continua al actualizar la AMI o la plantilla de lanzamiento. También puede usar un enfoque gradual, conocido como implementación canario, para probar una nueva AMI o plantilla de lanzamiento en un conjunto pequeño de instancias antes de implementarla para todo el grupo.

Enlaces de ciclo de vida

Los enlaces de ciclo de vida son útiles para definir acciones personalizadas que se invocan cuando se lanzan nuevas instancias o antes de que las terminen. Esta característica es especialmente útil para crear arquitecturas basadas en eventos, pero también le permite gestionar las instancias a lo largo de su ciclo de vida.

Compatibilidad con cargas de trabajo con estado

Los enlaces del ciclo de vida también ofrecen un mecanismo para mantener el estado de las instancias cuando se apagan. Para garantizar la continuidad de las aplicaciones con estado, también puede utilizar políticas personalizadas de protección contra la reducción horizontal o de terminación para evitar que las instancias con procesos de larga ejecución terminen anticipadamente.

Para obtener más información sobre los beneficios de Amazon EC2 Auto Scaling, consulte Beneficios del escalado automático para la arquitectura de aplicaciones.

Precios de Amazon EC2 Auto Scaling

Amazon EC2 Auto Scaling no conlleva cargos adicionales, por lo que es fácil probarlo y ver cómo puede beneficiar a AWS su arquitectura. Solo paga por los AWS recursos (por ejemplo, instancias EC2, volúmenes de EBS y CloudWatch alarmas) que utilice.

Introducción

Para empezar, complete el tutorial Cree su primer grupo de escalado automático para crear un grupo de escalado automático y ver cómo responde cuando termina una instancia de ese grupo.

Trabajo con grupos de Auto Scaling

Puede crear grupos de Auto Scaling, acceder a ellos y administrarlos con cualquiera de las siguientes interfaces:

- Consola de administración de AWS: proporciona una interfaz web que puede utilizar para acceder a los grupos de Auto Scaling. Si se ha registrado en uno Cuenta de AWS, puede acceder a sus grupos de Auto Scaling iniciando sesión en Consola de administración de AWS, utilizando el cuadro de búsqueda de la barra de navegación para buscar grupos de Auto Scaling y, a continuación, seleccionando grupos de Auto Scaling.
- AWS Command Line Interface (AWS CLI): proporciona comandos para un amplio conjunto de Servicios de AWS sistemas y es compatible con Windows, macOS y Linux. Para empezar, consulte Prepárese para usar el AWS CLI. Para obtener más información, consulte autoscaling en la Referencia de comandos de la AWS CLI .
- AWS Tools for Windows PowerShell— Proporciona comandos para un amplio conjunto de AWS productos para quienes escriben en el PowerShell entorno. Para empezar, consulte la Herramientas de AWS para PowerShell Guía del usuario de . Para obtener más información, consulte la Referencia de cmdlet de Herramientas de AWS para PowerShell.
- AWS SDK: proporciona operaciones de API específicas del idioma y se ocupa de muchos de los detalles de la conexión, como el cálculo de las firmas, la gestión de los reintentos de solicitudes y la gestión de los errores. Para obtener más información, consulte SDK de AWS.
- API de consulta: proporciona acciones de API de nivel bajo a las que se llama mediante solicitudes HTTPS. Utilizar la API de consulta es la forma más directa de obtener acceso a Servicios de AWS. Sin embargo, requiere que la aplicación gestione detalles de nivel inferior, como, por ejemplo, la

generación del hash para firmar la solicitud y la gestión de errores. Para obtener más información, consulte la Referencia de la API de Amazon EC2 Auto Scaling.

- CloudFormation— Soporta la creación de grupos de Auto Scaling mediante CloudFormation plantillas. Para obtener más información, consulte Cree grupos de Auto Scaling con AWS CloudFormation.

Para conectarse mediante programación a un dispositivo Servicio de AWS, utilice un punto final. Para obtener información sobre los puntos de enlace para las llamadas a Amazon EC2 Auto Scaling, consulte los puntos de enlace y las cuotas de Amazon EC2 Auto Scaling Referencia general de AWSen los puntos de enlace y

Beneficios del escalado automático para la arquitectura de aplicaciones

Añadir Amazon EC2 Auto Scaling a la arquitectura de sus aplicaciones es una forma de maximizar los beneficios de AWS la nube. Cuando se utiliza Amazon EC2 Auto Scaling, sus aplicaciones disfrutan de los siguientes beneficios:

- Mejor tolerancia a errores. Amazon EC2 Auto Scaling puede detectar cuándo una instancia está en mal estado, terminarla y lanzar una instancia para reemplazarla. También puede configurar Amazon EC2 Auto Scaling para que use varias zonas de disponibilidad. Si una zona de disponibilidad deja de estar disponible, Amazon EC2 Auto Scaling puede lanzar instancias en otra para compensar.
- Mejor disponibilidad. Amazon EC2 Auto Scaling puede ayudarle a garantizar que la aplicación tiene siempre la capacidad adecuada para gestionar la demanda de tráfico actual.
- Mejor administración de costos. Amazon EC2 Auto Scaling puede aumentar y reducir de forma dinámica la capacidad según sea necesario. Dado que se paga por las instancias EC2 que se utilizan, es posible ahorrar dinero lanzando instancias cuando se necesitan y terminándolas cuando ya no son necesarias.

Contenido

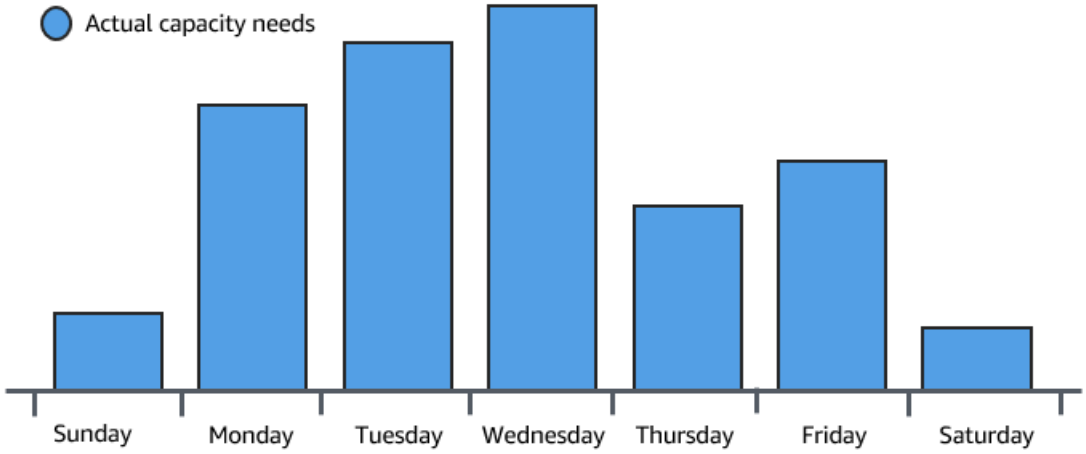
- Ejemplo: cubrir una demanda variable
- Ejemplo: Arquitectura de aplicaciones web
- Ejemplo: distribuir instancias entre zonas de disponibilidad

- Distribución de instancias
- Actividades de reequilibrio

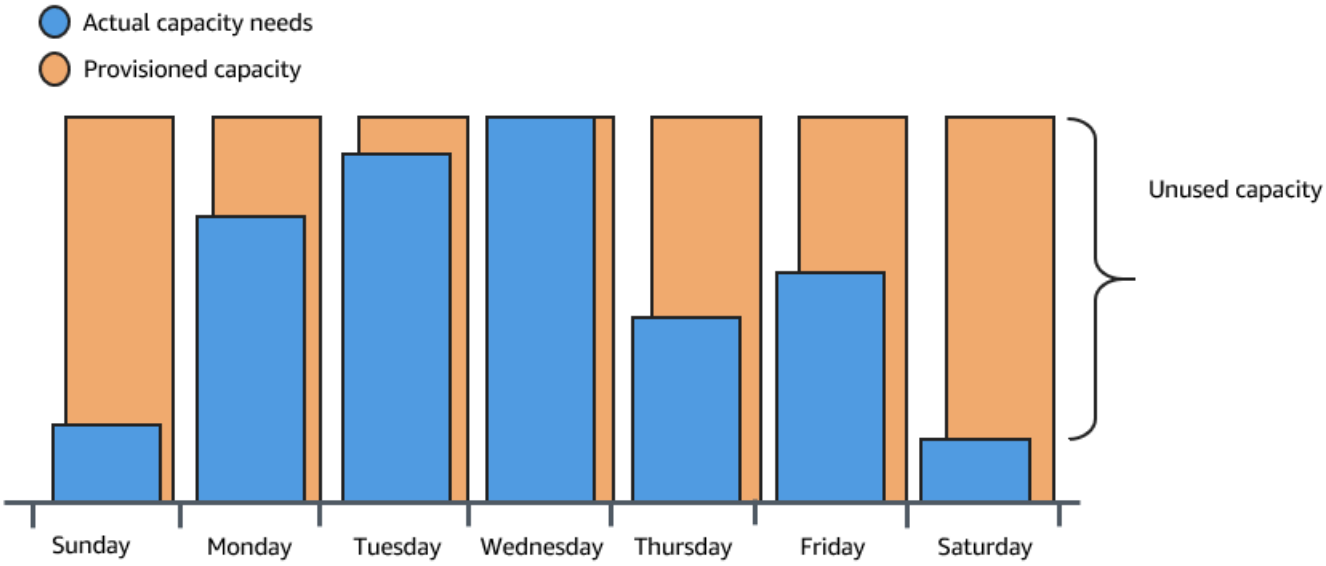
Ejemplo: cubrir una demanda variable

Para mostrar algunos de los beneficios de Amazon EC2 Auto Scaling, considere una aplicación web básica que se ejecuta en AWS. Esta aplicación permite a los empleados buscar salas de conferencias para las reuniones. Durante el comienzo y el final de la semana, el uso de esta aplicación es mínimo. A mitad de semana, hay más empleados que programan reuniones, de modo que la demanda de la aplicación aumenta de forma significativa.

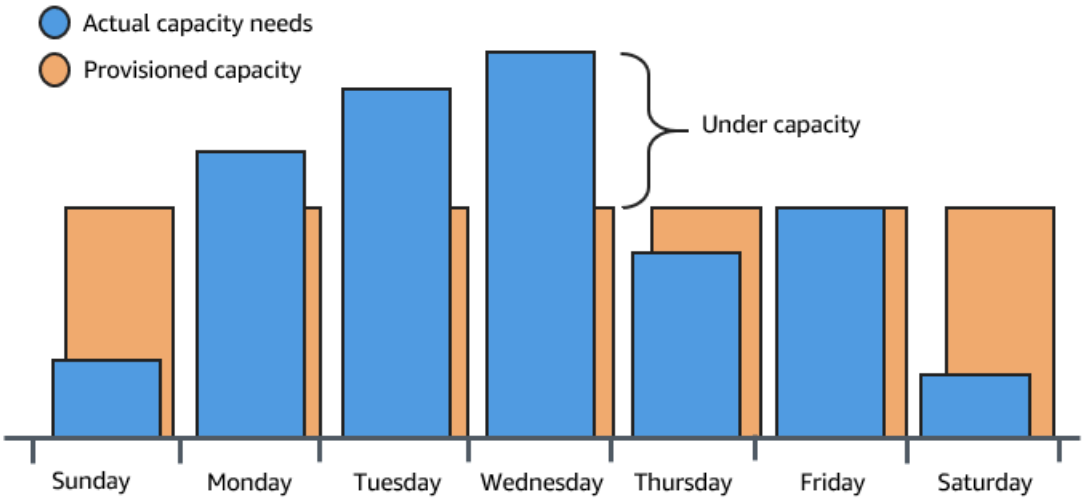
En el siguiente gráfico se muestra cuánta capacidad de la aplicación se usa a lo largo de una semana.



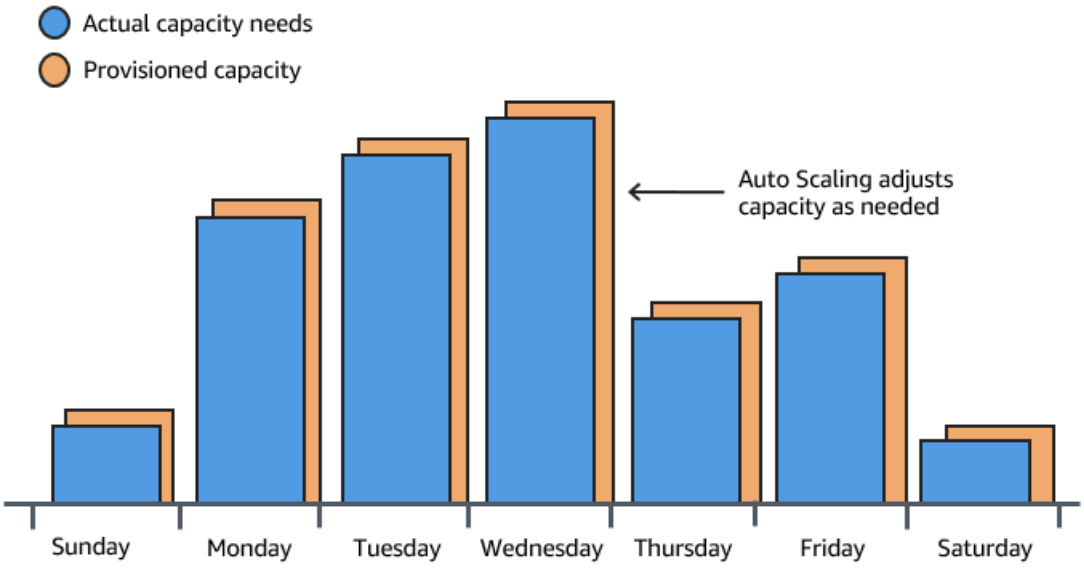
Tradicionalmente, hay dos formas de planificar estos cambios de capacidad. La primera opción consiste en agregar servidores suficientes para que la aplicación siempre tenga capacidad suficiente para satisfacer la demanda. La desventaja de esta opción, sin embargo, es que hay días en que la aplicación no necesita tanta capacidad. La capacidad adicional permanece sin utilizar y, en esencia, aumenta el costo de mantener la aplicación en ejecución.



La segunda opción es tener capacidad suficiente para gestionar la demanda media de la aplicación. Esta opción es menos cara, ya que no necesita comprar equipos que utilizará solo de vez en cuando. Sin embargo, existe el riesgo de que la experiencia del cliente se vea afectada si la demanda de la aplicación supera su capacidad.



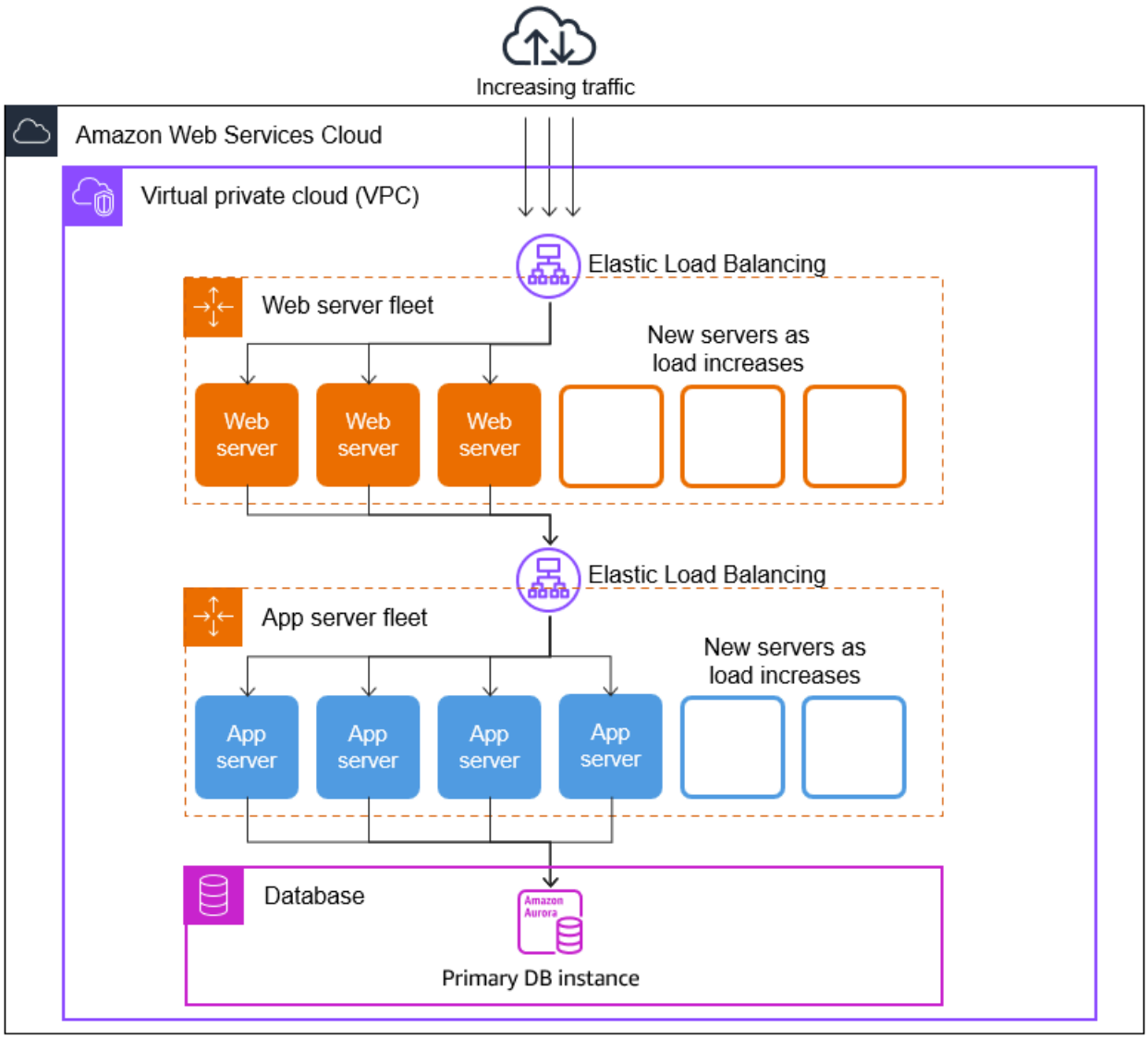
Añadiendo Amazon EC2 Auto Scaling a esta aplicación, dispone de una tercera opción. Puede añadir nuevas instancias a la aplicación solo cuando sea necesario y terminarlas cuando ya no las necesite. Como Amazon EC2 Auto Scaling utiliza instancias EC2, solo tiene que pagar por las instancias que utilice, cuando las utilice. Ahora tiene una arquitectura rentable que proporciona la mejor experiencia a los clientes a la vez que se minimizan los gastos.



Ejemplo: Arquitectura de aplicaciones web

En un escenario de aplicación web común, ejecuta varias copias de su aplicación de forma simultánea para cubrir el volumen del tráfico de sus clientes. Estas copias múltiples de la aplicación se alojan en instancias EC2 idénticas (servidores en la nube), cada una de las cuales tramita solicitudes de los clientes.

Amazon EC2 Auto Scaling administra el lanzamiento y la terminación de estas instancias EC2 en su nombre. Usted define un conjunto de criterios (como una CloudWatch alarma de Amazon) que determina cuándo el grupo de Auto Scaling lanza o termina las instancias de EC2. Añadir grupos de Auto Scaling a la arquitectura de red ayuda a aumentar la disponibilidad y la tolerancia a errores de la aplicación.



Puede crear todos los grupos de Auto Scaling que necesite. Por ejemplo, puede crear un grupo de escalado automático para cada capa.

Para distribuir el tráfico entre las instancias del grupo de escalado automático, puede introducir un balanceador de carga en su arquitectura. Para obtener más información, consulte Elastic Load Balancing.

Ejemplo: distribuir instancias entre zonas de disponibilidad

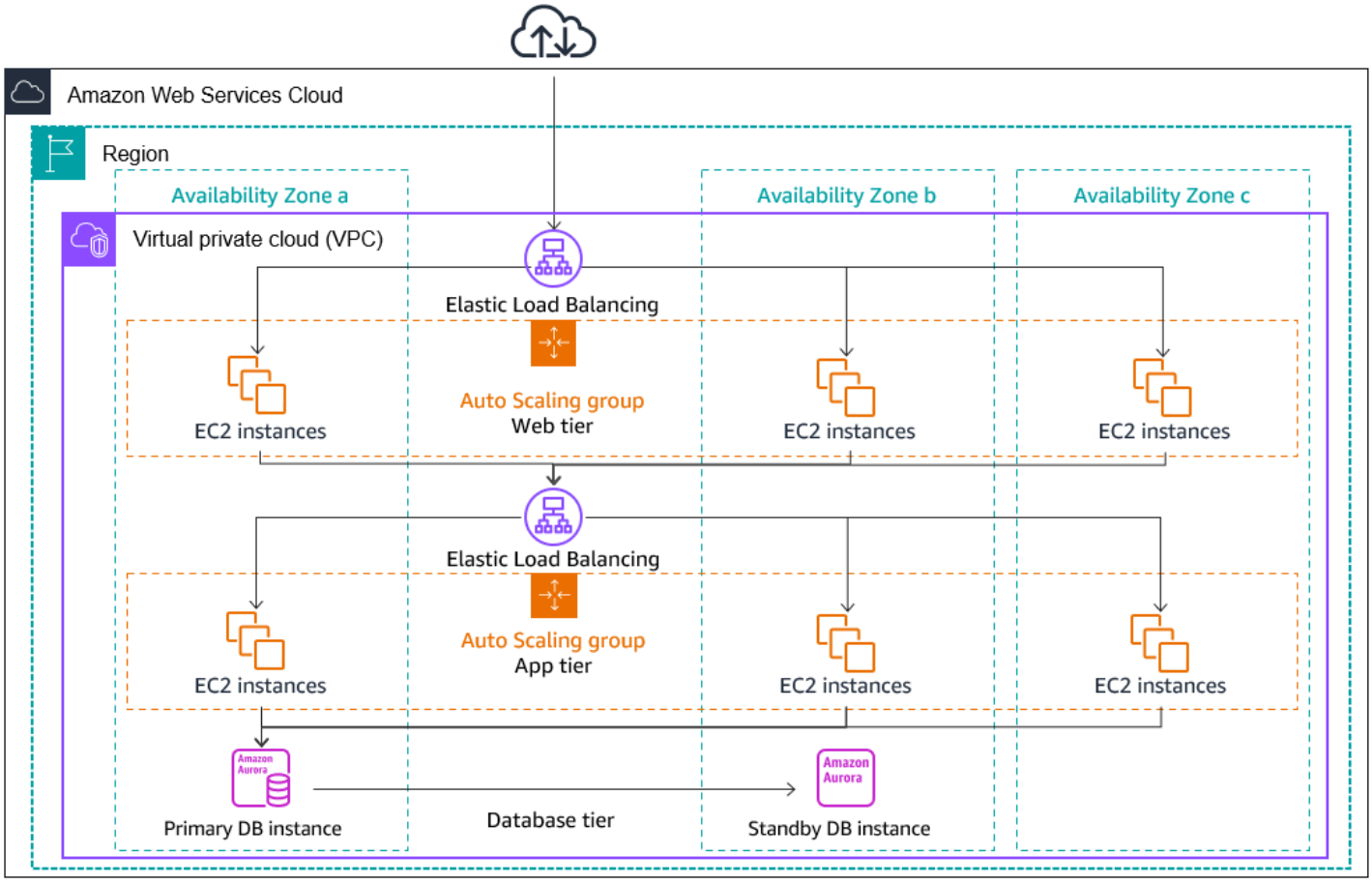
Las zonas de disponibilidad son ubicaciones aisladas en una Región de AWS. Cada región tiene varias zonas de disponibilidad diseñadas para proporcionar alta disponibilidad para la región. Las zonas de disponibilidad son independientes y, por lo tanto, se aumenta la disponibilidad de las

aplicaciones al diseñar la aplicación para que utilice varias zonas. Para obtener más información, consulte Resiliencia en Amazon EC2 Auto Scaling.

Una zona de disponibilidad se identifica mediante el Región de AWS código seguido de una letra identificadora (por ejemplo,`us-east-1a`). Si crea la VPC y las subredes en lugar de utilizar la VPC predeterminada, puede definir una o más subredes en cada zona de disponibilidad. Cada subred debe residir enteramente en una zona de disponibilidad y no puede abarcar otras zonas. Para obtener más información, consulte Cómo funciona Amazon VPC en la Guía del usuario de Amazon VPC.

Al crear un grupo de escalado automático, debe elegir la VPC y las subredes en las que implementará el grupo de escalado automático. Amazon EC2 Auto Scaling crea las instancias en las subredes elegidas. Así, cada instancia está asociada a una zona de disponibilidad específica elegida por Amazon EC2 Auto Scaling. Cuando se lanzan las instancias, Amazon EC2 Auto Scaling intenta distribuirlas de manera uniforme entre las zonas para lograr una alta disponibilidad y fiabilidad.

La siguiente imagen muestra una descripción general de la arquitectura de varios niveles implementada en tres zonas de disponibilidad.



Distribución de instancias

Amazon EC2 Auto Scaling intenta mantener automáticamente cantidades equivalentes de instancias en cada zona de disponibilidad habilitada. Para ello, Amazon EC2 Auto Scaling intenta lanzar nuevas instancias en la zona de disponibilidad con el menor número de instancias. Si hay varias subredes seleccionadas para una zona de disponibilidad, Amazon EC2 Auto Scaling intentará lanzar instancias en la subred que tiene la mayor cantidad de direcciones IP disponibles en la zona de disponibilidad. Sin embargo, si el intento fracasa, Amazon EC2 Auto Scaling intenta lanzar las instancias en otra zona de disponibilidad hasta que lo logra.

En los casos en los que una zona de disponibilidad deja de estar en buen estado o no está disponible, la distribución de instancias puede quedar de manera desigual entre las zonas de disponibilidad. Cuando se recupera la zona de disponibilidad, Amazon EC2 Auto Scaling reequilibra automáticamente el grupo de escalado automático. Para hacerlo, lanza instancias en las zonas de disponibilidad habilitadas con menos instancias y terminando instancias en otros lugares.

Actividades de reequilibrio

Las actividades de reequilibrio se dividen en dos categorías: reequilibrio de zona de disponibilidad y reequilibrio de capacidad.

Reequilibrio de zona de disponibilidad

Cuando se producen determinadas acciones, el grupo de escalado automático puede quedar desequilibrado entre las zonas de disponibilidad. Amazon EC2 Auto Scaling compensa este desequilibrio reequilibrando las zonas de disponibilidad. Las siguientes acciones pueden derivar en una actividad de reequilibrio:

- Usted cambia las zonas de disponibilidad asociadas a su grupo de escalado automático.
- Termina o desasocia instancias de forma explícita, o las coloca en espera y entonces el grupo queda desequilibrado.
- Una zona de disponibilidad que anteriormente tenía capacidad insuficiente se recupera y ahora tiene capacidad adicional.
- Una zona de disponibilidad que anteriormente tenía un precio de spot superior a su precio máximo ahora tiene un precio de spot inferior a su precio máximo.

Al reequilibrar, Amazon EC2 Auto Scaling lanza nuevas instancias antes de terminar las más antiguas. De esta manera, el reequilibrio no pone en peligro el rendimiento ni la disponibilidad de la aplicación.

Como Amazon EC2 Auto Scaling intenta lanzar nuevas instancias antes de terminar las anteriories, si se está en la capacidad máxima especificada o cerca de ella podría impedir o detener completamente las actividades de reequilibrio.

Para evitar este problema, el sistema puede superar temporalmente la capacidad máxima especificada de un grupo durante una actividad de reequilibrio. De modo predeterminado, puede hacerlo con un margen del 10 por ciento o en una instancia, lo que sea mayor. El margen solo se amplía si el grupo está en su capacidad máxima, o cerca de ella, y necesita un reequilibrio. La extensión se mantiene solamente mientras sea necesaria para reequilibrar el grupo (normalmente unos minutos).

Como alternativa, puede establecer umbrales para un grupo de escalado automático mediante una política de mantenimiento de instancias, y el grupo solo puede aumentar o disminuir la capacidad dentro de ese rango de umbrales. De esta forma, puede controlar la rapidez con la que su grupo se reequilibra. Para obtener más información, consulte Políticas de mantenimiento de instancias.

Reequilibrio de la capacidad

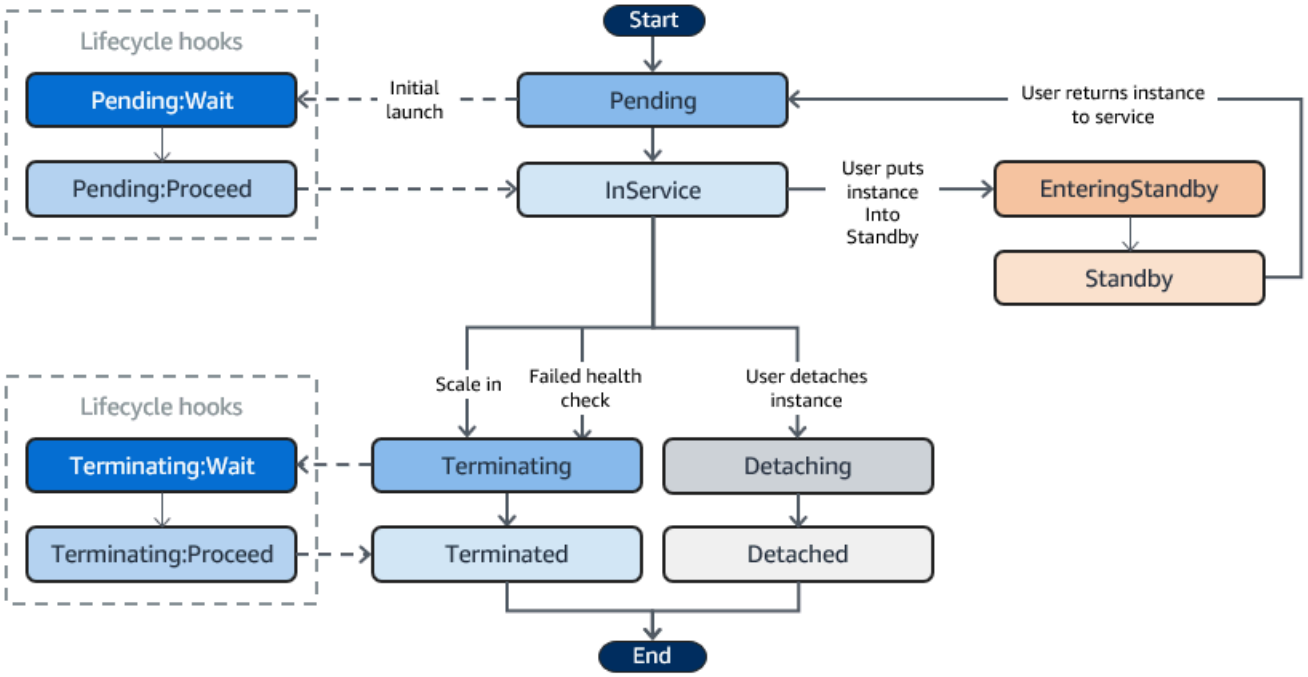
Puede habilitar el reequilibrio de capacidad para los grupos de escalado automático cuando utilice instancias de spot. Esto permite que Amazon EC2 Auto Scaling intente lanzar una instancia de spot siempre que Amazon EC2 notifica que una instancia de spot tiene un riesgo elevado de interrupción. Después de lanzar una nueva instancia, termina una instancia anterior. Para obtener más información, consulte Reequilibrio de la capacidad en Auto Scaling para reemplazar las instancias de spot en riesgo.

Ciclo de vida de instancias de Amazon EC2 Auto Scaling

Las instancias EC2 de un grupo de escalado automático tienen una ruta o ciclo de vida que difiere de las de otras instancias EC2. El ciclo de vida comienza cuando el grupo de escalado automático lanza una instancia y la pone en servicio. El ciclo de vida finaliza cuando el usuario termina la instancia o el grupo de escalado automático retira la instancia del servicio y la termina.

> **Note**
>
> Las instancias se cobran en cuanto se lanzan, incluido el tiempo en que aún no están en servicio.

La siguiente ilustración muestra las transiciones entre los estados de la instancia en el ciclo de vida de Amazon EC2 Auto Scaling.



Escalado ascendente

Los siguientes eventos de escalado horizontal indican al grupo de escalado automático que lance instancias EC2 y las asocie al grupo:

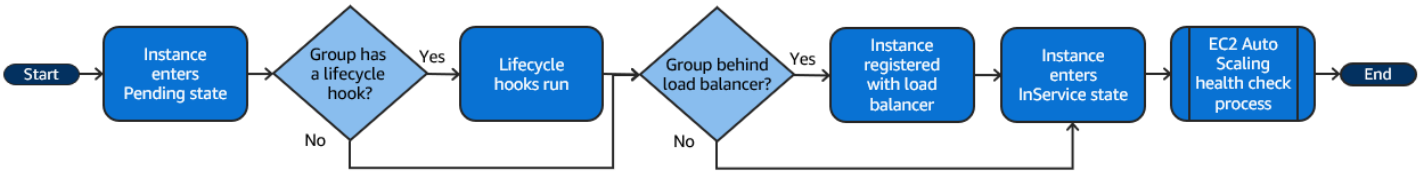
- Aumenta manualmente el tamaño del grupo. Para obtener más información, consulte Cambio de la capacidad deseada del grupo de escalado automático.
- Crea una política de escalado para aumentar automáticamente el tamaño del grupo en función del aumento de la demanda especificado. Para obtener más información, consulte Escalado dinámico para Amazon EC2 Auto Scaling.
- Configura el escalado basado en programación para aumentar el tamaño del grupo en un momento determinado. Para obtener más información, consulte Escalado programado para Amazon EC2 Auto Scaling.

Cuando se produce un evento de escalado horizontal, el grupo de escalado automático lanza el número necesario de instancias de EC2, con su plantilla de lanzamiento asignada. Estas instancias comienzan en el estado `Pending`. Si agrega un enlace de ciclo de vida al grupo de escalado automático, puede realizar una acción personalizada aquí. Para obtener más información, consulte Enlaces de ciclo de vida.

Cuando cada instancia está totalmente configurada y supera las comprobaciones de estado de Amazon EC2, se asocia al grupo de escalado automático y pasa a tener el estado `InService`. La instancia se tendrá en cuenta para calcular la capacidad deseada del grupo de escalado automático.

Si su grupo de escalado automático está configurado para recibir tráfico de un equilibrador de carga de Elastic Load Balancing, Amazon EC2 Auto Scaling registra automáticamente su instancia con este equilibrador antes de marcar la instancia como `InService`.

A continuación, se resumen los pasos necesarios para registrar una instancia con un equilibrador de carga para un evento de escalado horizontal.



Instancias en servicio

Las instancias permanecen en el estado `InService` hasta que se produce alguna de las siguientes situaciones:

- Se produce un evento de reducción horizontal y Amazon EC2 Auto Scaling decide terminar esta instancia para reducir el tamaño del grupo de escalado automático. Para obtener más información, consulte Control de las instancias de Auto Scaling que se terminan durante una reducción horizontal.
- Coloca la instancia en estado `Standby`. Para obtener más información, consulte Entrada y salida del modo de espera.
- Desconecta la instancia del grupo de escalado automático. Para obtener más información, consulte Desconexión o asociación de instancias en el grupo de escalado automático.
- La instancia no supera el número necesario de comprobaciones de estado, por lo que se elimina del grupo de escalado automático, se termina y se reemplaza. Para obtener más información, consulte Comprobaciones de estado para instancias en un grupo de escalado automático.

Reducción horizontal

Los siguientes eventos de reducción horizontal indican al grupo de escalado automático que desconecte las instancias EC2 del grupo y las termine:

- Reduce manualmente el tamaño del grupo. Para obtener más información, consulte Cambio de la capacidad deseada del grupo de escalado automático.
- Crea una política de escalado para reducir automáticamente el tamaño del grupo en función de la reducción de la demanda especificada. Para obtener más información, consulte Escalado dinámico para Amazon EC2 Auto Scaling.
- Configura el escalado basado en programación para reducir el tamaño del grupo en un momento determinado. Para obtener más información, consulte Escalado programado para Amazon EC2 Auto Scaling.

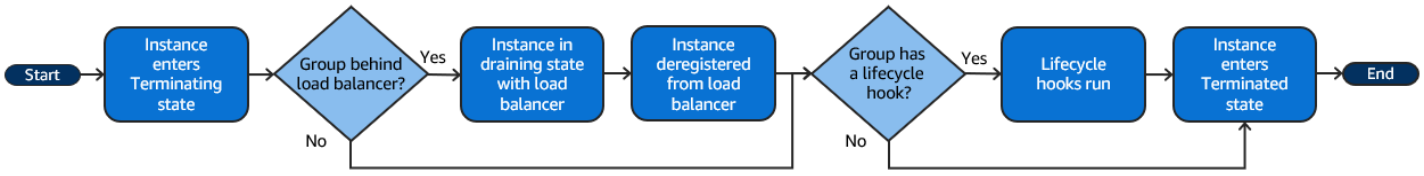
Es importante que, cada vez que cree un evento de reducción horizontal, cree también un evento de escalado horizontal correspondiente. De esta forma, se asegurará de que los recursos asignados a la aplicación coinciden lo máximo posible con la demanda de esos recursos.

Cuando se produce un evento de reducción horizontal, el grupo de escalado automático termina una o varias instancias. El grupo de escalado automático utiliza su política de terminación para determinar qué instancias debe terminar. Las instancias que se encuentran en proceso de terminación del grupo de escalado automático adoptan el estado `Terminating` y no se pueden volver a poner en servicio.

Si su grupo de escalado automático está configurado para recibir tráfico de un equilibrador de carga de Elastic Load Balancing, Amazon EC2 Auto Scaling cancela automáticamente el registro de la instancia que termina del equilibrador de carga. Al anular el registro de la instancia, se garantiza que todas las solicitudes nuevas se redirijan a otras instancias del grupo de destino del equilibrador de carga, mientras que se permite que las conexiones existentes a la instancia continúen hasta que venza el retraso de cancelación del registro.

Si agrega un enlace de ciclo de vida al grupo de escalado automático, puede realizar una acción personalizada en la instancia que termina. Para obtener más información, consulte Enlaces de ciclo de vida. Por último, la instancia se termina completamente y adopta el estado `Terminated`.

A continuación, se resumen los pasos necesarios para anular el registro de una instancia con un equilibrador de carga para un evento de reducción horizontal.



Desconexión de una instancia

Puede desconectar una instancia del grupo de escalado automático. Una vez que la instancia se ha desconectado, puede administrarla por separado del grupo de escalado automático o asociarla a otro grupo de escalado automático.

Para obtener más información, consulte Desconexión o asociación de instancias en el grupo de escalado automático.

Asociación de una instancia

Puede asociar una instancia EC2 en ejecución que cumpla determinados criterios al grupo de escalado automático. Una vez que la instancia se ha asociado, se administra como parte del grupo de escalado automático.

Para obtener más información, consulte Desconexión o asociación de instancias en el grupo de escalado automático.

Enlaces de ciclo de vida

Puede agregar un enlace de ciclo de vida al grupo de escalado automático para realizar acciones personalizadas cuando las instancias se lanzan y se terminan.

Cuando Amazon EC2 Auto Scaling responde a un evento de escalado horizontal, lanza una o varias instancias. Estas instancias comienzan en el estado `Pending`. Si ha agregado un enlace de ciclo de vida `autoscaling:EC2_INSTANCE_LAUNCHING` al grupo de escalado automático, las instancias pasan del estado `Pending` al estado `Pending:Wait`. Una vez completada la acción de ciclo de vida, las instancias adoptan el estado `Pending:Proceed`. Cuando las instancias están totalmente configuradas, se asocian al grupo de escalado automático y adoptan el estado `InService`.

Cuando Amazon EC2 Auto Scaling responde a un evento de reducción horizontal, termina una o varias instancias. Estas instancias se desconectan del grupo de escalado automático y adoptan el estado `Terminating`. Si ha agregado un enlace de ciclo de vida

`autoscaling:EC2_INSTANCE_TERMINATING` al grupo de escalado automático, las instancias pasan del estado `Terminating` al estado `Terminating:Wait`. Una vez completada la acción de ciclo de vida, las instancias adoptan el estado `Terminating:Proceed`. Cuando las instancias se terminan completamente, adoptan el estado `Terminated`.

Para obtener más información, consulte Enlaces de ciclo de vida de Amazon EC2 Auto Scaling.

Entrada y salida del modo de espera

Puede poner cualquier instancia que tenga un estado `InService` en estado `Standby`. Esto le permite retirar la instancia del servicio, solucionar un problema o realizar cambios en ella y ponerla de nuevo en servicio.

Las instancias con un estado `Standby` siguen estando administradas por el grupo de escalado automático. Sin embargo, no son parte activa de la aplicación hasta que las pone de nuevo en servicio.

Para obtener más información, consulte Eliminación temporal de las instancias de un grupo de escalado automático.

Cuotas para los recursos y grupos de escalado automático

Cuenta de AWS Tiene cuotas predeterminadas, anteriormente denominadas límites, para cada AWS servicio. A menos que se indique lo contrario, cada cuota es Region-specific. Puede solicitar el aumento de algunas cuotas, pero otras no se pueden aumentar.

Para consultar las cuotas de Amazon EC2 Auto Scaling, abra la consola de Service Quotas. En el panel de navegación, elija Servicios de AWS y seleccione Amazon EC2 Auto Scaling.

Para solicitar un aumento de cuota, consulte Solicitud de aumento de cuota en la Guía del usuario de Service Quotas. Si la cuota aún no se encuentra disponible en Service Quotas, utilice el formulario de aumento del límite de Auto Scaling. Los aumentos de cuota están asociados a la región para la que se solicitan.

Recursos de Amazon EC2 Auto Scaling

Cuenta de AWS Tiene las siguientes cuotas relacionadas con la cantidad de grupos de Auto Scaling y configuraciones de lanzamiento que puede crear.

Recurso	Cuota predeterminada
Grupos de Auto Scaling por región	500
Configuraciones de lanzamiento por región	200

Configuración del grupo de escalado automático

Cuenta de AWS Tiene las siguientes cuotas relacionadas con la configuración de los grupos de Auto Scaling. No es posible cambiarlos.

Recurso	Cuota
Políticas de escalado por grupo de escalado automático	50
Acciones programadas por grupo de escalado automático	125
Ajustes de pasos por política de escalado de pasos	20
Enlaces de ciclo de vida por grupo de escalado automático	50
Temas de SNS por grupo de escalado automático	10
Equilibradores de carga clásicos por grupo de escalado automático	50
Grupos de destino de equilibrador de carga de Elastic Load Balancing por grupo de escalado automático	50
Grupos de destino de VPC Lattice por grupo de escalado automático	5

Operaciones de API de los grupos de Auto Scaling

Amazon EC2 Auto Scaling proporciona operaciones de API para realizar cambios en los grupos de Auto Scaling por lotes. A continuación se indican los límites de API para el número máximo de elementos (miembros máximos de la matriz) permitidos en una sola operación. No es posible cambiarlos.

Operación	Número máximo de miembros de la matriz
AttachInstances	20 ID de instancia
AttachLoadBalancers	10 equilibradores de carga
AttachLoadBalancerTargetGroups	10 grupos de destino
BatchDeleteScheduledAction	50 acciones programadas
BatchPutScheduledUpdateGroupAction	50 acciones programadas
DetachInstances	20 ID de instancia
DetachLoadBalancers	10 equilibradores de carga
DetachLoadBalancerTargetGroups	10 grupos de destino
EnterStandby	20 ID de instancia
ExitStandby	20 ID de instancia
SetInstanceProtection	50 ID de instancia

Solicitud de limitación para la API de Amazon EC2 Auto Scaling

Las solicitudes a la API de Amazon EC2 Auto Scaling se limitan mediante un esquema de bucket de token para mantener el ancho de banda del servicio. Para obtener más información, consulte API request rate en la Referencia de la API de Amazon EC2 Auto Scaling.

Tasas de terminación de EC2

Amazon EC2 Auto Scaling determina de forma dinámica el número de operaciones de terminación de instancia EC2 que puede realizar en un momento en el que el grupo de escalado automático se reduzca horizontalmente. Esto significa que podría ver variaciones en el número de instancias terminadas a la vez entre los grupos de escalado automático. Estas variaciones se deben a

consideraciones externas, como si Amazon EC2 Auto Scaling debe anular el registro de las instancias con un equilibrador de carga.

Otros servicios

Las cuotas de otros servicios, como Amazon EC2 y Amazon VPC, pueden afectar sus grupos de escalado automático. Puede utilizarlas Service Quotas para actualizar las cuotas de las instancias de EC2 y otros recursos de su Cuenta de AWS. En la Service Quotas consola, puede ver todas las cuotas de servicio disponibles y solicitar su aumento. Para obtener más información, consulte Requesting a quota increase (Solicitud de un aumento de cuota) en la Guía del usuario de Service Quotas .

Para obtener información sobre las cuotas específicas de las plantillas de lanzamiento, consulte Restricciones de las plantillas de lanzamiento en la Guía del usuario de Amazon EC2.

Configuración para usar Amazon EC2 Auto Scaling

Antes de empezar a utilizar Amazon EC2 Auto Scaling, realice las siguientes tareas.

Tareas

- Prepárese para usar el AWS CLI

Prepárese para usar el AWS CLI

Puede utilizar las herramientas de línea de AWS comandos para emitir comandos en la línea de comandos de su sistema para realizar Amazon EC2 Auto Scaling y otras AWS tareas.

Para usar el AWS Command Line Interface (AWS CLI), descargue, instale y configure la versión 1 o 2 del AWS CLI. La misma funcionalidad de Amazon EC2 Auto Scaling está disponible en las versiones 1 y 2. Para instalar la versión 1 de AWS CLI , consulte Instalar, actualizar y desinstalar AWS CLI en la Guía del usuario de la versión 1 de AWS CLI . Para instalar la AWS CLI versión 2, consulte Instalación o actualización de la última versión de AWS CLI en la Guía del usuario de la AWS CLI versión 2.

AWS CloudShell le permite omitir la instalación de AWS CLI en su entorno de desarrollo y utilizarla en Consola de administración de AWS su lugar. Además de evitar la instalación, tampoco es necesario configurar las credenciales ni especificar una región. Su Consola de administración de AWS sesión proporciona este contexto a AWS CLI. Puede usarlo de AWS CloudShell forma compatible Regiones de AWS. Para obtener más información, consulte Cree grupos de Auto Scaling desde la línea de comandos usando AWS CloudShell.

Para obtener más información, consulte autoscaling en la Referencia de comandos de la AWS CLI .

Introducción a Amazon EC2 Auto Scaling

Para empezar con Amazon EC2 Auto Scaling, puede seguir los tutoriales que le presentan el servicio.

Temas

- Tutorial: Creación de su primer grupo de escalado automático
- Tutorial: Configuración de una aplicación con escalado y balanceo de carga aplicados

Para obtener más tutoriales que se centran en herramientas específicas para administrar el ciclo de vida de las instancias de un grupo de escalado automático, consulte los siguientes temas:

- Tutorial: Configuración de un enlace de ciclo de vida que invoca una función Lambda. En este tutorial, se muestra cómo usar Amazon EventBridge para crear reglas que invoquen funciones de Lambda en función de los eventos que ocurren en las instancias de su grupo de Auto Scaling.
- Tutorial: Uso del script de datos y los metadatos de la instancia para recuperar el estado del ciclo de vida. En este tutorial, se muestra cómo usar el Servicio de metadatos de instancias (IMDS) para invocar una acción desde la propia instancia.

Antes de crear un grupo de escalado automático para usarlo con una aplicación, revísela a fondo mientras se pone en marcha en la nube de Nube de AWS. Considere lo siguiente:

- Entre cuántas zonas de disponibilidad debe distribuirse el grupo de Auto Scaling.
- Qué recursos existentes se pueden usar, como grupos de seguridad o Amazon Machine Images (AMIs).
- Si desea escalar para aumentar o disminuir la capacidad, o solo desea asegurarse de que siempre haya un número específico de servidores en funcionamiento. Tenga en cuenta que Amazon EC2 Auto Scaling puede hacer ambas cosas a la vez.
- Qué métricas son más relevantes para el rendimiento de la aplicación.
- Cuánto tiempo tarda en inicializar y aprovisionar un servidor.

Cuanto mejor conozca su aplicación, mayor será la eficacia de su arquitectura de Auto Scaling.

Tutorial: Creación de su primer grupo de escalado automático

Este tutorial proporciona una introducción práctica a Amazon EC2 Auto Scaling a través del. Consola de administración de AWS Creará una plantilla de inicialización que defina sus instancias de EC2 y un grupo de escalado automático con una sola instancia. Después de inicializar su grupo de escalado automático, finalizará la instancia y verificará que se haya eliminado del servicio y se haya sustituido. Para mantener un número constante de instancias, Amazon EC2 Auto Scaling detecta y responde automáticamente a las comprobaciones de estado y accesibilidad de Amazon EC2.

Cuando se registre AWS, podrá empezar a utilizar Amazon EC2 Auto Scaling de forma gratuita mediante AWS la capa gratuita. Puede usar la capa gratuita para iniciar y usar una instancia `t2.micro` de forma gratuita durante 12 meses (en regiones donde `t2.micro` no esté disponible, puede usar una instancia `t3.micro` de la capa gratuita). Si inicia una instancia que no está dentro de la capa gratuita, se le cobrará la tarifa de uso estándar de Amazon EC2 por la instancia. Para obtener más información, consulte Precios de Amazon EC2.

Tareas

- Prepararse para el tutorial
- Paso 1: crear una plantilla de inicialización
- Paso 2: Crear un grupo de Auto Scaling de instancia única
- Paso 3: Verificar el grupo de Auto Scaling
- Paso 4: Terminar una instancia en el grupo de Auto Scaling
- Paso 5: Siguientes pasos
- Paso 6: limpiar

Prepararse para el tutorial

En esta explicación se presupone que está familiarizado con el lanzamiento de instancias EC2 y que ya ha creado un par de claves y un grupo de seguridad.

Para comenzar a usar Amazon EC2 Auto Scaling, puede utilizar la VPC predeterminada para su Cuenta de AWS. La VPC predeterminada incluye una subred pública predeterminada en cada zona de disponibilidad y una puerta de enlace de Internet asociada a la VPC. Puede verla VPCs en la VPCspágina Su de la consola Amazon Virtual Private Cloud (Amazon VPC).

Paso 1: crear una plantilla de inicialización

En este paso, creará una plantilla de inicialización que especifica el tipo de instancia de EC2 que Amazon EC2 Auto Scaling crea automáticamente. Incluya información como el ID de la Amazon Machine Image (AMI) que se va a usar, el tipo de instancia, el par de claves y los grupos de seguridad.

Para crear una plantilla de lanzamiento

1. Abra la consola de Amazon EC2 y vaya a la página Plantillas de inicialización.
2. En la barra de navegación superior, debe seleccionar una Región de AWS. La plantilla de lanzamiento y el grupo de escalado automático que cree están vinculados a la región que especifica.
3. Elija Crear plantilla de inicialización.
4. Para Launch template name (Nombre de plantilla de lanzamiento), ingrese **`my-template-for-auto-scaling`**.
5. En Auto Scaling guidance (Guía de Auto Scaling), seleccione la casilla de verificación.
6. En Application and OS Images (Amazon Machine Image) (Imágenes de aplicación y SO [imagen de máquina de Amazon]), elija una versión de Amazon Linux 2 (HVM) en la lista Quick Start (Inicio rápido). La (AMI) sirve de plantilla de configuración básica para sus instancias.
7. En Instance type (Tipo de instancia), elija una configuración de hardware que sea compatible con la AMI que ha especificado.
8. (Opcional) Para Key pair (login) (Par de claves [inicio de sesión]), elija un par de claves existente. Los pares de claves se utilizan durante la conexión SSH a una instancia de Amazon EC2. La conexión a una instancia no se incluye como parte de este tutorial. Por lo tanto, no tiene que especificar un par de claves, a menos que tenga la intención de conectarse a la instancia mediante SSH.
9. En Network settings (Configuración de red), expanda Advanced network configuration (Configuración avanzada de red) y proceda del modo siguiente:

    a. Elija Add network interface (Agregar interfaz de red) para configurar la interfaz de red principal.

    b. Para Asignar automáticamente una IP pública, especifique si la instancia recibe una dirección IPv4 pública. De forma predeterminada, Amazon EC2 asigna una dirección IPv4 pública si la instancia EC2 se lanza en una subred predeterminada o si la instancia se lanza

en una subred que se ha configurado para asignar automáticamente una dirección pública. IPv4 Si no necesita conectarse a su instancia, elija Deshabilitar.

c. Para ID de grupos de seguridad, elija un grupo de seguridad en la misma VPC que piensa utilizar como VPC para su grupo de escalado automático. Si no especifica ningún grupo de seguridad, la instancia se asocia automáticamente al grupo de seguridad predeterminado de la VPC.

d. Para Eliminar al terminar, elija Sí para eliminar la interfaz de red, cuando se elimina la instancia.

10. Elija Crear plantilla de inicialización.
11. En la página de confirmación, seleccione Create Auto Scaling group (Crear grupo de Auto Scaling.

Paso 2: Crear un grupo de Auto Scaling de instancia única

Utilice el siguiente procedimiento para continuar donde lo dejó después de crear la plantilla de inicialización.

Para crear un grupo de Auto Scaling

1. En la página Choose launch template or configuration (Elegir una plantilla de lanzamiento o configuración), para Auto Scaling group name (Nombre de grupo de Auto Scaling), ingrese **`my-first-asg`**.
2. Elija Siguiente.

   Aparece la página Elegir las opciones de inicialización de instancias, que le permite elegir la configuración de red de la VPC que desea que utilice el grupo de escalado automático y le ofrece opciones para inicializar instancias bajo demanda e instancias de spot.
3. En la sección Red, mantenga la VPC configurada como la VPC predeterminada que haya elegido o seleccione su propia Región de AWS VPC. La VPC predeterminada se configura automáticamente para proporcionar conectividad a Internet a la instancia. Esta VPC incluye una subred pública en cada zona de disponibilidad de la región.
4. En Availability Zones and subnets (Zonas de disponibilidad y subredes), elija una subred de cada zona de disponibilidad que desee incluir. Utilice subredes en varias zonas de disponibilidad para lograr una alta disponibilidad. Para obtener más información, consulte Consideraciones a la hora de elegir subredes de VPC.

5. En la sección Instance type requirements (Requisitos del tipo de instancia), utilice la configuración predeterminada para simplificar este paso. (No anule la plantilla de lanzamiento). En este tutorial, solo lanzará una instancia bajo demanda con el tipo de instancia especificado en la plantilla de lanzamiento.
6. Mantenga el resto de los valores predeterminados para este tutorial y elija Skip to review (Omitir para revisar).

   > **Note**
   >
   > El tamaño inicial del grupo está determinado por su capacidad deseada. El valor predeterminado es instancia 1.

7. En la página Review (Revisar), revise la información del grupo y elija Auto Scaling group (Grupo de Auto Scaling).

Paso 3: Verificar el grupo de Auto Scaling

Ahora que ha creado su grupo de Auto Scaling, está listo para verificar si el grupo ha lanzado una instancia EC2.

> **Tip**
>
> En el siguiente procedimiento, observará las secciones Activity history (Historial de actividad) e Instances (Instancias) del grupo de Auto Scaling. En ambas, ya deberían aparecer las columnas con nombre. Para mostrar las columnas ocultas o cambiar el número de filas que aparecen, elija el icono de engranaje en la esquina superior derecha de cada sección para abrir el modal de preferencias, actualice la configuración según sea necesario y seleccione Confirm (Confirmar).

Para verificar si el grupo de Auto Scaling ha lanzado una instancia EC2

1. Abra la página grupos de escalado automático en la consola de Amazon EC2.
2. Seleccione la casilla de verificación junto al grupo de Auto Scaling que acaba de crear.

   Se abre un panel dividido en la parte inferior de la página Auto Scaling groups (Grupos de Auto Scaling). La primera pestaña disponible es la pestaña Details (Detalles) que muestra información sobre el grupo de Auto Scaling.

3. Seleccione la segunda pestaña, Activity (Actividad). En Activity history (Historial de actividad), puede ver el progreso de las actividades que están asociadas al grupo de Auto Scaling. La columna Status (Estado) muestra el estado actual de su instancia. Mientras se está lanzando la instancia, la columna de estado muestra `Not yet in service`. El estado cambia a `Successful` cuando se lanza la instancia. También puede utilizar el botón de actualización para ver el estado actual de la instancia.
4. En la pestaña Instance management (Administración de instancia), en Instances (Instancias), puede ver el estado de la instancia.
5. Compruebe que la instancia se ha lanzado correctamente. La instancia tarda poco tiempo en lanzarse.

   - La columna Lifecycle (Ciclo de vida) muestra el estado de su instancia. Al principio, la instancia tiene el estado `Pending`. Cuando una instancia está lista para recibir tráfico, su estado es `InService`.
   - La columna Estado muestra el resultado de las comprobaciones de estado de Amazon EC2 Auto Scaling correspondientes a su instancia.

Paso 4: Terminar una instancia en el grupo de Auto Scaling

Utilice estos pasos para obtener más información sobre cómo funciona Amazon EC2 Auto Scaling, específicamente, cómo lanza nuevas instancias cuando sea necesario. El tamaño mínimo del grupo de Auto Scaling creado en este aprendizaje es una instancia. Por lo tanto, si termina la instancia en ejecución, Amazon EC2 Auto Scaling debe lanzar una instancia nueva para sustituirla.

1. Abra la página grupos de escalado automático en la consola de Amazon EC2.
2. Seleccione la casilla situada junto al grupo de escalado automático.
3. En la pestaña Instance management (Administración de instancias), en Instances (Instancias), seleccione el ID de la instancia.

   Tras ello, accederá a la página Instances (Instancias)) de la consola de Amazon EC2, donde puede terminar la instancia.
4. Elija Actions (Acciones), Instance State (Estado de la instancia), Terminate (Terminar). Cuando se le pida confirmación, elija Yes, Terminate.
5. En el panel de navegación, seleccione Auto Scaling y elija Auto Scaling Groups (Grupos de Auto Scaling). Seleccione el grupo de Auto Scaling y elija la pestaña Activity (Actividad).

Al finalizar una instancia desde la página Instancias, transcurren uno o dos minutos antes de que se inicialice una nueva instancia. En el historial de actividad, cuando comience la actividad de escalado, verá una entrada para la terminación de la primera instancia y una entrada para el lanzamiento de una nueva instancia. Use el botón de actualización hasta que vea las nuevas entradas.

6. En la pestaña Instance management (Administración de instancias), la sección Instances (Instancias) muestra solo la nueva instancia.
7. En el panel de navegación, en Instances (Instancias), elija Instances. Esta página muestra la instancia terminada y la nueva instancia en ejecución.

Paso 5: Siguientes pasos

Vaya al paso siguiente si desea eliminar la infraestructura básica que acaba de crear. De lo contrario, puede utilizar esta infraestructura como punto de partida y realizar alguna de las siguientes operaciones:

- Conéctese a la instancia de Linux mediante el Administrador de sesiones o SSH. Para obtener más información, consulte Conectarse a la instancia de EC2 mediante el Administrador de sesiones y Conectarse a la instancia de Linux con SSH en la Guía del usuario de Amazon EC2.
- Configure una notificación de Amazon SNS para recibir una notificación cada vez que el grupo de escalado automático lance o termine instancias. Para obtener más información, consulte Opciones de notificaciones de Amazon SNS.
- Escale manualmente el grupo de escalado automático para probar la notificación de SNS. Para obtener más información, consulte Cambio de la capacidad deseada de su grupo de escalado automático.

Además, para comenzar a familiarizarse con los conceptos de escalado automático, puede leer sobre Políticas de escalado de seguimiento de destino. Si la carga de la aplicación cambia, el grupo de escalado automático puede escalarse horizontalmente (agregar instancias) y reducirse horizontalmente (ejecutar menos instancias) automáticamente si se ajusta la capacidad deseada del grupo entre los límites de capacidad mínimo y máximo. Para obtener más información sobre cómo ajustar estos límites, consulte Establecimiento de límites de escalado para el grupo de escalado automático.

Paso 6: limpiar

Puede eliminar la infraestructura de escalado o eliminar únicamente el grupo de escalado automático y conservar la plantilla de inicialización para usarla en otro momento.

Si ha lanzado una instancia que no está dentro del nivel gratuito de AWS, debe terminar la instancia para evitar cargos adicionales. Cuando termine la instancia, los datos asociados con ella también se eliminarán.

Para eliminar el grupo de Auto Scaling

1. Abra la página grupos de escalado automático en la consola de Amazon EC2.
2. Seleccione la casilla situada junto al grupo de escalado automático (`my-first-asg`).
3. Elija Eliminar.
4. Cuando se le pida la confirmación, escriba **`delete`** para confirmar la eliminación del grupo de escalado automático especificado y, a continuación, elija Delete (Eliminar).

   Un icono de carga en la columna Name (Nombre) indica que el grupo de Auto Scaling se está eliminando. Una vez eliminado, las columnas Deseadas, Mín y Máx muestran instancias de `0` para el grupo de escalado automático. Se tarda unos minutos en terminar la instancia y eliminar el grupo. Actualice la lista para ver el estado actual.

Omita el procedimiento siguiente si desea mantener su plantilla de lanzamiento.

Para eliminar la plantilla de lanzamiento

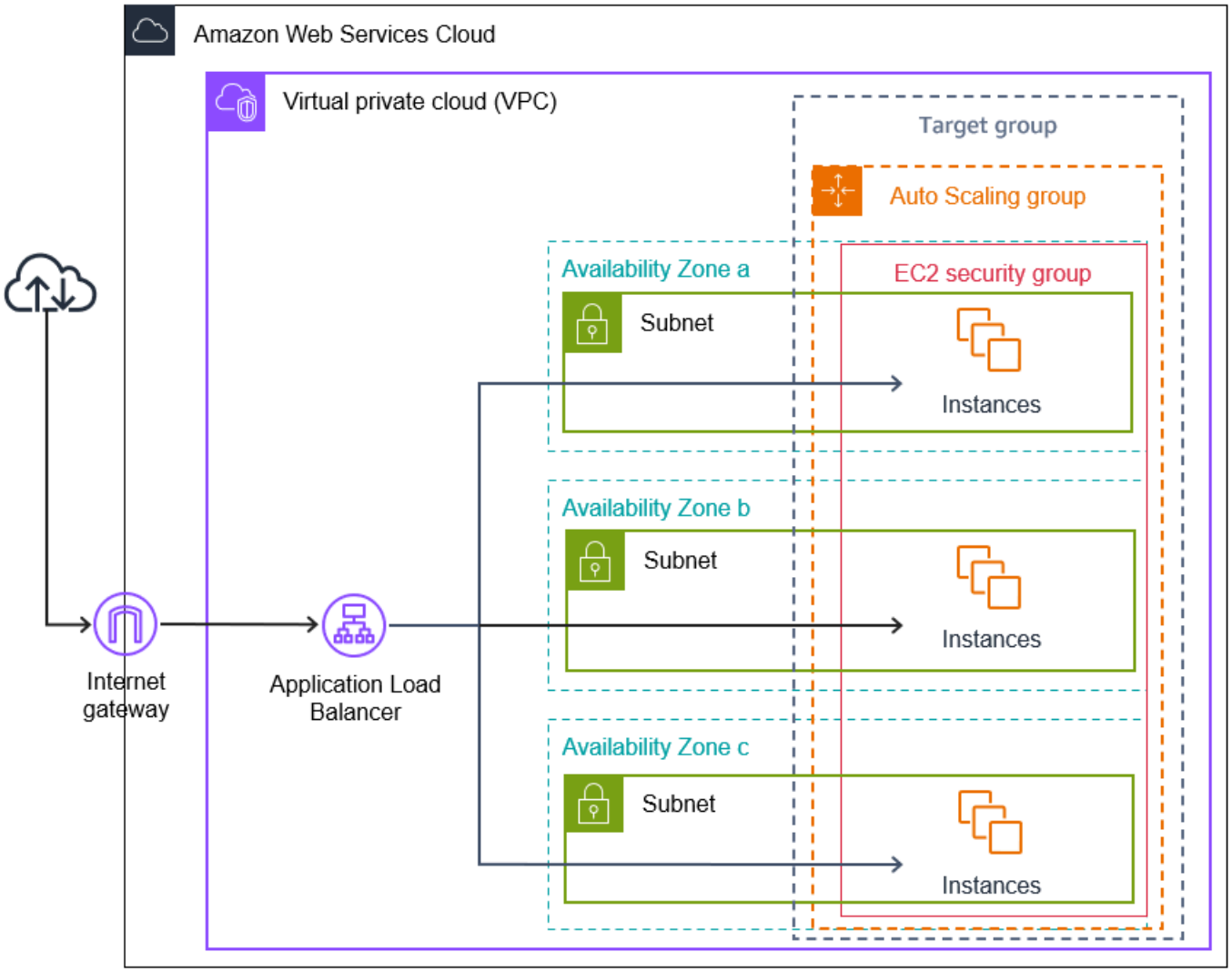
1. Abra la página Launch templates (Plantillas de lanzamiento) de la consola de Amazon EC2.
2. Seleccione la plantilla de lanzamiento (`my-template-for-auto-scaling`).
3. Elija Actions, Delete template.
4. Cuando se le pida la confirmación, escriba **`Delete`** para confirmar la eliminación de la plantilla de lanzamiento especificada y, a continuación, elija Delete (Eliminar).

Tutorial: Configuración de una aplicación con escalado y balanceo de carga aplicados

> **Important**
>
> Antes de explorar este tutorial, le recomendamos que consulte la siguiente guía introductoria: Creación de su primer grupo de escalado automático.

Registrar el grupo de Auto Scaling con un balanceador de carga Elastic Load Balancing ayuda a configurar una aplicación con balanceo de carga. Elastic Load Balancing funciona con Amazon EC2 Auto Scaling para distribuir el tráfico entrante entre las instancias de Amazon EC2 en buen estado. Esto aumenta la escalabilidad y disponibilidad de la aplicación. Puede habilitar Elastic Load Balancing dentro de varias zonas de disponibilidad para aumentar la tolerancia a errores de sus aplicaciones.

En este tutorial, tratamos los pasos básicos para configurar una aplicación con balanceo de carga al crear el grupo de Auto Scaling. Cuando haya finalizado, la arquitectura debe ser similar a la del diagrama siguiente:



Elastic Load Balancing admite distintos tipos de equilibradores de carga. Le recomendamos que utilice un Application Load Balancer para este tutorial.

Para obtener más información sobre cómo ingresar un balanceador de carga en la arquitectura, consulte Utilice Elastic Load Balancing para distribuir el tráfico entrante de la aplicación en su grupo de escalado automático..

Tareas

- Requisitos previos
- Paso 1: Configurar una plantilla de lanzamiento o una configuración de lanzamiento
- Paso 2: Crear un grupo de Auto Scaling)
- Paso 3: Verificar que el balanceador de carga está adjunto
- Paso 4: Siguientes pasos

- Paso 5: Eliminar
- Recursos relacionados

Requisitos previos

- Un balanceador de carga y grupo de destino. Asegúrese de elegir las mismas zonas de disponibilidad para el balanceador de carga que tiene previsto utilizar para su grupo de Auto Scaling. Para obtener más información, consulte Introducción a Elastic Load Balancing en la Guía del usuario de Elastic Load Balancing.
- Un grupo de seguridad para la plantilla o configuración de lanzamiento. El grupo de seguridad debe permitir el acceso desde el balanceador de carga en el puerto del agente de escucha (normalmente el puerto 80 para tráfico HTTP) y el puerto que desea que utilice Elastic Load Balancing para comprobaciones de estado. Para obtener más información, consulte la documentación aplicable:
  - Grupos de seguridad de destino en la Guía del usuario de balanceadores de carga de aplicaciones
  - Grupos de sguridad de destino en la Guía del usuario de balanceadores de carga de red

  Opcionalmente, si las instancias tendrán direcciones IP públicas, puede permitir tráfico SSH si necesita conectarse a las instancias.
- (Opcional) Un rol de IAM al que se le otorga acceso a AWS la aplicación.
- (Opcional) Una Amazon Machine Image (AMI) definida como plantilla de origen de las instancias de Amazon EC2. Para crear una, lance una instancia. Especifique el rol de IAM (si ha creado uno) y todos los scripts de configuración que necesite como datos de los usuarios. Conéctese a la instancia y personalícela. Por ejemplo, puede instalar software y aplicaciones, copiar datos y adjuntar volúmenes de EBS adicionales. Pruebe la aplicación en la instancia para asegurarse de que se ha configurado correctamente. Guarde esta configuración actualizada como una AMI personalizada. Puede terminar la instancia si no la necesita más tarde. Las instancias que se lancen desde la nueva AMI personalizada incluirán todas las configuraciones que definió al crearla.
- Una nube virtual privada (VPC). Este tutorial hace referencia a la VPC predeterminada, pero puede utilizar una propia. Si utiliza supropia VPC, asegúrese de que tiene una subred asignada a cada zona de disponibilidad de la región de en la que esté trabajando. Como mínimo, debe tener dos subredes públicas disponibles para crear el balanceador de carga. También debe tener dos subredes privadas o dos subredes públicas para crear el grupo de Auto Scaling y registrarlo en el balanceador de carga.

Paso 1: Configurar una plantilla de lanzamiento o una configuración de lanzamiento

Utilice una plantilla de lanzamiento o una configuración de lanzamiento para este tutorial.

Temas

- Creación o selección de una plantilla de inicialización
- Seleccionar o crear una configuración de lanzamiento

Creación o selección de una plantilla de inicialización

Si ya tiene una plantilla de lanzamiento que desee utilizar, selecciónela con el siguiente procedimiento.

Para seleccionar una plantilla de lanzamiento existente

1. Abra la página Launch templates (Plantillas de lanzamiento) de la consola de Amazon EC2.
2. En la barra de navegación de la parte superior de la pantalla, elija la región donde se creó el balanceador de carga.
3. Seleccione una plantilla de lanzamiento.
4. Elija Actions (Acciones), Create an Auto Scaling group (Crear un grupo de Auto Scaling).

También puede crear una nueva plantilla de lanzamiento mediante el siguiente procedimiento.

Para crear una plantilla de lanzamiento

1. Abra la página Launch templates (Plantillas de lanzamiento) de la consola de Amazon EC2.
2. En la barra de navegación de la parte superior de la pantalla, elija la región donde se creó el balanceador de carga.
3. Elija Crear plantilla de inicialización.
4. Escriba un nombre y una descripción para la versión inicial de la plantilla de lanzamiento.
5. En Application and OS Images (Amazon Machine Image) (Imágenes de aplicaciones y SO [imagen de máquina de Amazon]), elija el ID de la AMI para las instancias. Puede buscar en todas las AMI disponibles o seleccionar una AMI en la lista Recents (Recientes) o Quick Start (Inicio rápido). Si no ve la AMI que necesita, elija Browse more AMIs (Buscar más AMI) para navegar por el catálogo completo de AMI.

6. En Instance type, seleccione una configuración de hardware de sus instancias que sea compatible con la AMI que ha especificado.
7. (Opcional) En Key pair (login) (Par de claves [inicio de sesión]), elija el par de claves que va a usar al conectarse a sus instancias.
8. En Network settings (Configuración de red), expanda Advanced network configuration (Configuración avanzada de red) y proceda del modo siguiente:

   a. Elija Add network interface (Agregar interfaz de red) para configurar la interfaz de red principal.

   b. En el caso de la IP Auto-assign pública, especifique si las instancias reciben direcciones IPv4 públicas. De forma predeterminada, Amazon EC2 asigna direcciones IPv4 públicas si la instancia de EC2 se lanza en una subred predeterminada o si la instancia se inicia en una subred configurada para asignar de manera automática una dirección IPv4 pública. Si no necesita conectarse a las instancias, puede elegir Desactivar para evitar que las instancias del grupo reciban tráfico directamente desde internet. En este caso, recibirán tráfico solo desde el balanceador de carga.

   c. Para Security group ID (ID de grupo de seguridad), especifique un grupo de seguridad para las instancias de la misma VPC que el balanceador de carga.

   d. Para Delete on termination (Eliminar al terminar), elija Yes (Sí). Esto elimina la interfaz de red cuando el grupo de Auto Scaling escala, y termina la instancia a la que la interfaz de red está asociada.
9. (Opcional) Para distribuir de forma segura credenciales a las instancias, en Advanced details (Detalles avanzados), IAM instance profile (Perfil de instancia de IAM), escriba el Nombre de recurso de Amazon (ARN) de su rol de IAM.
10. (Opcional) Para especificar los datos de usuario o un script de configuración para las instancias, pegue los datos o el script en Advanced details, User data.
11. Elija Crear plantilla de inicialización.
12. En la página de confirmación, elija Create Auto Scaling group (Crear grupo de Auto Scaling).

Seleccionar o crear una configuración de lanzamiento

> **Note**
>
> Se desaconseja encarecidamente utilizar configuraciones de inicialización en aplicaciones nuevas, porque es una característica heredada sin inversión planificada. Además, las

cuentas nuevas que se crearon el 1 de junio de 2023 o después no tendrán la opción de crear nuevas configuraciones de inicialización mediante la consola. Para obtener más información, consulte Configuraciones de inicialización de Auto Scaling.

Para seleccionar una configuración de lanzamiento existente

1. Abra la página Launch configurations (Configuraciones de lanzamiento) de la consola de Amazon EC2.
2. En la barra de navegación de la parte superior, elija la región donde se creó el equilibrador de carga.
3. Seleccione una configuración de lanzamiento.
4. Elija Actions (Acciones), Create an Auto Scaling group (Crear un grupo de Auto Scaling).

También puede crear una nueva configuración de lanzamiento mediante el siguiente procedimiento.

Para crear una configuración de lanzamiento

1. Abra la página Launch configurations (Configuraciones de lanzamiento) de la consola de Amazon EC2. Cuando se le pida confirmación, elija Ver configuraciones de lanzamiento para confirmar que desea ver la página Configuraciones de lanzamiento.
2. En la barra de navegación de la parte superior, elija la región donde se creó el equilibrador de carga.
3. Elija Crear una configuración de lanzamiento, e ingrese un nombre para la configuración de lanzamiento.
4. Para Amazon Machine Image (AMI), ingrese el ID de la AMI de sus instancias como criterio de búsqueda.
5. En el paso Instance Type (Tipo de instancias), seleccione la configuración de hardware de la instancia.
6. En Additional configuration (Configuración adicional), preste atención a los campos siguientes:

    a. (Opcional) Para distribuir de forma segura credenciales a la instancia EC2, para perfil de instancias de IAM, seleccione su rol de IAM. Para obtener más información, consulte Rol de IAM para aplicaciones que se ejecuten en instancias de Amazon EC2.

b. (Opcional) Para especificar los datos de usuario o un script de configuración para la instancia, pegue los datos o el script en Advanced details (Detalles avanzados), User data (Datos de usuario.

c. (Opcional) En Advanced Details (Detalles avanzados), IP Address Type (Tipo de dirección IP), mantenga el valor predeterminado. Al crear el grupo de Auto Scaling, puede asignar una dirección IP pública a instancias del grupo de Auto Scaling mediante subredes que tengan habilitado el atributo de direccionamiento de IP públicas, como las subredes predeterminadas de la VPC predeterminada. Alternativamente, si no necesita conectarse a sus instancias, puede elegir Do not assign a public IP address to any instances (No asignar una dirección IP pública a ninguna instancia) para evitar que las instancias del grupo reciban tráfico directamente desde internet. En este caso, recibirán tráfico solo desde el balanceador de carga.

7. Para Grupos de seguridad, elija un grupo de seguridad existente de la misma VPC que el balanceador de carga. Si mantiene la opción Create a new security group (Crear un nuevo grupo de seguridad) seleccionada, se configura una regla SSH predeterminada para instancias Amazon EC2 que ejecutan Linux. Se configura una regla de RDP predeterminada para instancias de Amazon EC2 que ejecutan Windows.

8. Para Key pair (login) (Par de claves [inicio de sesión]), elija una opción en Key pair options (Opciones de par de claves).

   Si ya ha configurado un par de claves de la instancia de Amazon EC2, puede elegirlo aquí.

   Si aún no tiene un par de claves de instancia de Amazon EC2, elija Create a new key pair (Crear un nuevo par de claves) y asígnele un nombre fácil de reconocer. Elija Download Key Pair (Descargar par de claves) para descargar el par de claves en su equipo.

   > **Important**
   >
   > No elija Proceed without a key pair (Continuar si un par de claves) si necesita establecer conexión con las instancias.

9. Seleccione la casilla de confirmación y, a continuación, elija Create launch configuration.

10. Seleccione el cuadro de verificación situada junto al nombre de la nueva configuración de lanzamiento y elija Actions (Acciones), Create Auto Scaling Group (Crear grupo de Auto Scaling).

Paso 2: Crear un grupo de Auto Scaling)

Utilice el siguiente procedimiento para continuar donde lo dejó después de crear o seleccionar la plantilla o la configuración de lanzamiento.

Para crear un grupo de Auto Scaling

1. En la página Choose launch template or configuration (Elegir una plantilla o configuración de lanzamiento), para el nombre del grupo de Auto Scaling, ingrese un nombre para su grupo de Auto Scaling.
2. [Solo plantilla de lanzamiento] En Launch template (Plantilla de lanzamiento), elija si el grupo de Auto Scaling utiliza el valor predeterminado, la última versión o una versión específica de la plantilla de lanzamiento para escalado horizontal.
3. Elija Siguiente.

   Aparece la página Elegir opciones de lanzamiento de instancias, que le permite elegir la configuración de red de VPC que desea que use el grupo de Auto Scaling y le ofrece opciones para lanzar On-Demand e instancias puntuales (si eligió una plantilla de lanzamiento).
4. En la sección Network (Red), en VPC, elija la VPC que haya utilizado para el equilibrador de carga. Si elige la VPC predeterminada, se configura automáticamente para proporcionar conectividad a Internet a las instancias. Esta VPC incluye una subred pública en cada zona de disponibilidad de la región.
5. En Availability Zones and subnets (Zonas de disponibilidad y subredes), elija una o más subredes de cada zona de disponibilidad que desee incluir, en función de las zonas de disponibilidad en las que se encuentre el equilibrador de carga. Para obtener más información, consulte Consideraciones a la hora de elegir subredes de VPC.
6. [Solo plantilla de lanzamiento] En la sección Instance type requirements (Requisitos del tipo de instancia), utilice la configuración predeterminada para simplificar este paso. (No anule la plantilla de lanzamiento). En este tutorial, solo lanzará On-Demand instancias con el tipo de instancia especificado en la plantilla de lanzamiento.
7. Elija Next (Siguiente) para ir a la página Configure advanced options (Configuración de opciones avanzadas).
8. En la sección Load balancing (Equilibrador de carga), elija Attach to an existing load balancer (Adjuntar a un equilibrador de carga existente) para adjuntar el grupo a un equilibrador de carga ya existente. Puede elegir Choose from your load balancer target groups (Elegir entre los grupos de destino del equilibrador de carga) o Choose from Classic Load Balancers (Elegir entre los

Classic Load Balancer). A continuación, puede elegir el nombre de un grupo de destino para el Application Load Balancer o el Network Load Balancer que creó, o bien elegir el nombre de un Classic Load Balancer.

9. (Opcional) En Comprobaciones de estado, Tipos de comprobaciones de estado adicionales, seleccione Activar las comprobaciones de estado de Elastic Load Balancing.
10. (Opcional) En Período de gracia de comprobación de estado, ingrese el tiempo, en segundos. Este es la cantidad de tiempo que Amazon EC2 Auto Scaling debe esperar antes de comprobar el estado de una instancia una vez que pasa al estado `InService`. Para obtener más información, consulte Establezca el periodo de gracia de la comprobación de estado para un grupo de escalado automático.
11. Cuando haya terminado de configurar el grupo de Auto Scaling, elija Skip to review (Omitir para revisar).
12. En la página Review (Revisar), revise los detalles del grupo de Auto Scaling. Si desea realizar cambios, haga clic en Edit. Cuando termine, elija Create Auto Scaling group (Crear grupo de Auto Scaling).

Después de crear el grupo de Auto Scaling con el balanceador de carga asociado, el balanceador de carga registra automáticamente nuevas instancias a medida que se conectan. Solo tiene una instancia en este punto, por lo que no hay mucho que registrar. Sin embargo, puede agregar más instancias actualizando la capacidad deseada del grupo. Para obtener instrucciones paso a paso, consulte Cambio de la capacidad deseada de su grupo de escalado automático.

Paso 3: Verificar que el balanceador de carga está adjunto

Para verificar que el balanceador de carga está adjunto

1. Desde la página Auto Scaling groups (Grupos de Auto Scaling) de la consola de Amazon EC2, seleccione la casilla de verificación situada junto al grupo de Auto Scaling.
2. En la pestaña Details (Detalles), en Load balancing (Balanceador de carga), se muestran los grupos de destino del balanceador de carga asociado o balanceadores de carga clásicos.
3. En la pestaña Activity (Actividad) en Activity history (Historial de actividad), puede comprobar que las instancias se hayan lanzado correctamente. La columna Status (Estado) indica si el grupo de Auto Scaling ha lanzado las instancias correctamente. Si las instancias no se lanzan, puede encontrar ideas de solución de problemas para problemas de lanzamiento de instancias comunes en Solución de problemas de Amazon EC2 Auto Scaling.

4. En la pestaña Instance management (Administración de instancias), en Instances (Instancias), puede comprobar que las instancias estén listas para recibir tráfico. Inicialmente, las instancias están en estado `Pending`. Cuando una instancia está lista para recibir tráfico, su estado es `InService`. La columna Health status (Estado) muestra el resultado de la comprobación de estado de Amazon EC2 Auto Scaling correspondiente a su instancia. Aunque una instancia pueda estar marcada en buen estado, el balanceador de carga solo enviará tráfico a instancias que pasen las comprobaciones de estado del balanceador de carga.
5. Verifique que las instancias estén registradas en el balanceador de carga. Abra la página Grupos de destino de la consola de Amazon EC2. Seleccione el grupo de destino y elija la pestaña Targets (Destinos). Si el estado de las instancias es `initial`, es probable que se deba a que todavía están en proceso de registrarse o están siendo sometidas a comprobaciones de estado. Cuando el estado de las instancias sea `healthy`, están listas para utilizarse.

Paso 4: Siguientes pasos

Ahora que ha completado este tutorial, puede obtener más información:

- Amazon EC2 Auto Scaling determina si una instancia está en buen estado en función de las comprobaciones de estado que utiliza su grupo de escalado automático. Si habilita las comprobaciones de estado del equilibrador de carga y una instancia no supera las comprobaciones de estado, el grupo de escalado automático considera que la instancia no está en buen estado y la sustituye. Para obtener más información, consulte Comprobaciones de estado.
- Puede ampliar la aplicación a una zona de disponibilidad adicional de la misma región para aumentar la tolerancia a errores en caso de interrupción del servicio. Para obtener más información, consulte Agregar una zona de disponibilidad.
- Puede configurar el grupo de Auto Scaling para que utilice una política de escalado de seguimiento de destino. Esto aumenta o disminuye automáticamente el número de instancias a medida que cambie la demanda de las instancias. Esta permite que el grupo gestione los cambios en la cantidad de tráfico que recibe la aplicación. Para obtener más información, consulte Políticas de escalado de seguimiento de destino.

Paso 5: Eliminar

Cuando haya acabado con los recursos que creó para este tutorial, debería considerar la posibilidad de eliminarlos para evitar incurrir en gastos innecesarios.

Para eliminar el grupo de Auto Scaling

1. Abra la página grupos de escalado automático en la consola de Amazon EC2.
2. Seleccione la casilla situada junto al grupo de escalado automático.
3. Elija Eliminar.
4. Cuando se le pida la confirmación, escriba **`delete`** para confirmar la eliminación del grupo de escalado automático especificado y, a continuación, elija Delete (Eliminar).

   Un icono de carga en la columna Name (Nombre) indica que el grupo de Auto Scaling se está eliminando. Una vez eliminado, las columnas Deseadas, Mín y Máx muestran instancias de `0` para el grupo de escalado automático. Se tarda unos minutos en terminar la instancia y eliminar el grupo. Actualice la lista para ver el estado actual.

Omita el procedimiento siguiente si desea mantener su plantilla de lanzamiento.

Para eliminar la plantilla de lanzamiento

1. Abra la página Launch templates (Plantillas de lanzamiento) de la consola de Amazon EC2.
2. Seleccione la plantilla de lanzamiento.
3. Elija Actions, Delete template.
4. Cuando se le pida la confirmación, escriba **`Delete`** para confirmar la eliminación de la plantilla de lanzamiento especificada y, a continuación, elija Delete (Eliminar).

Omita el procedimiento siguiente si desea mantener su configuración de lanzamiento.

Para eliminar su configuración de lanzamiento

1. Abra la página Launch configurations (Configuraciones de lanzamiento) de la consola de Amazon EC2.
2. Seleccione la configuración de lanzamiento.
3. Seleccione Actions, Delete launch configuration.
4. Cuando se le pida confirmación, seleccione Eliminar.

Omita el siguiente procedimiento si desea mantener el balanceador de carga para usarlo en el futuro.

Para eliminar el equilibrador de carga

1. Abra la página de Load Balancers (Balanceadores de carga) en la consola de Amazon EC2.
2. Seleccione el balanceador de carga y elija Actions (Acciones), Delete (Eliminar).
3. Cuando se le indique que confirme, seleccione Yes, Delete (Sí, borrar).

Para eliminar los grupos de destino

1. Abra la página Grupos de destino de la consola de Amazon EC2.
2. Elija el grupo de destino y elija Actions (Acciones), Delete (Eliminar).
3. Cuando se le indique que confirme, seleccione Yes, Delete (Sí, borrar).

Recursos relacionados

Con CloudFormationél, puede crear y aprovisionar despliegues de AWS infraestructura de forma predecible y repetitiva, mediante el uso de archivos de plantilla para crear y eliminar un conjunto de recursos juntos como una sola unidad (una pila). Para obtener más información, consulte la Guía del usuario de AWS CloudFormation.

Para ver un tutorial que muestra cómo usar una plantilla de pila para aprovisionar un grupo de escalado automático y un Equilibrador de carga de aplicación, consulte Tutorial: Creación de una aplicación con escalado y equilibrio de carga en la Guía del usuario de AWS CloudFormation . Utilice el tutorial y la plantilla de ejemplo como punto de partida para crear plantillas similares para satisfacer sus necesidades.

Plantillas de lanzamiento de escalado automático

Una plantilla de lanzamiento es similar a una configuración de lanzamiento, ya que sirve para especificar la información de configuración de las instancias. Incluye el ID de la Amazon Machine Image (AMI), el tipo de instancia, un par de claves, los grupos de seguridad y el resto de los parámetros que se utilizan para lanzar instancias EC2. No obstante, la definición de una plantilla de lanzamiento en lugar de una configuración de lanzamiento le permite tener varias versiones de una plantilla de lanzamiento.

Con el control de versiones de plantillas de lanzamiento, puede crear un subconjunto del conjunto completo de parámetros. A continuación, puede reutilizarlo para crear otras versiones de la misma plantilla de lanzamiento. Por ejemplo, puede crear una plantilla de lanzamiento que defina una configuración base sin un script de datos de usuario o AMI. Después de crear la plantilla de lanzamiento, puede crear una nueva versión y agregar la AMI y los datos de usuario que tiene la versión más reciente de la aplicación para pruebas. Esto da como resultado dos versiones de la plantilla de lanzamiento. Almacenar una configuración base le ayuda a mantener los parámetros de configuración generales requeridos. Puede crear una nueva versión de su plantilla de lanzamiento desde la configuración base siempre que lo desee. También puede eliminar las versiones utilizadas para probar su aplicación cuando ya no las necesite.

Le recomendamos que utilice plantillas de lanzamiento para asegurarse de que está accediendo a las últimas características y mejoras. No todas las funciones de Amazon EC2 Auto Scaling están disponibles cuando se utilizan configuraciones de lanzamiento. Por ejemplo, no puede crear un grupo de Auto Scaling que lance tanto Spot como On-Demand Instances o que especifique varios tipos de instancias. Debe utilizar una plantilla de lanzamiento para configurar estas características. Para obtener más información, consulte Grupos de Auto Scaling con varios tipos de instancia y opciones de compra.

Con las plantillas de lanzamiento, también puede utilizar características más recientes de Amazon EC2. Esto incluye los parámetros (ID de AMI) de Systems Manager, la generación actual de volúmenes de IOPS aprovisionadas de EBS (io2), el etiquetado de volúmenes de EBS, las instancias T2 ilimitadas, reservas de capacidad, Capacity Blocks y hosts dedicados, por nombrar algunos.

Al crear una plantilla de lanzamiento, todos los parámetros son opcionales. Sin embargo, si una plantilla de lanzamiento no especifica una AMI, no puede agregar la AMI al crear el grupo de Auto Scaling. Si especifica una AMI pero ningún tipo de instancia, puede agregar uno o más tipos de instancia al crear el grupo de Auto Scaling.

Contenido

- Permisos para trabajar con plantillas de lanzamiento
- Operaciones de la API compatibles con plantillas de lanzamiento
- Creación de una plantilla de lanzamiento para un grupo de Auto Scaling
- Creación de una plantilla de lanzamiento mediante la configuración avanzada
- Migración de sus grupos de escalado automático a plantillas de lanzamiento
- Migración AWS CloudFormation pilas para lanzar plantillas
- Ejemplos de creación y administración de plantillas de lanzamiento con AWS CLI
- Uso AWS Systems Manager parámetros en lugar de ID de AMI en las plantillas de lanzamiento

Permisos para trabajar con plantillas de lanzamiento

En los procedimientos de esta sección se da por hecho que ya tiene los permisos necesarios para crear plantillas de lanzamiento. Para obtener información sobre cómo le concede permisos un administrador, consulte Controle el acceso a las plantillas de inicialización con permisos de IAM en la Guía del usuario de Amazon EC2.

Tenga en cuenta que, si no tiene suficientes permisos para usar y crear recursos especificados en una plantilla de lanzamiento, recibirá un error que indica que no está autorizado a utilizar la plantilla de lanzamiento cuanto intente especificarla para un grupo de escalado automático. Para obtener más información, consulte Solución de problemas de Amazon EC2 Auto Scaling: plantillas de lanzamiento.

Para ver ejemplos de políticas de IAM que permiten llamar a las operaciones de API `CreateAutoScalingGroup`, `UpdateAutoScalingGroup` y `RunInstances` con una plantilla de lanzamiento, consulte Control del uso de la plantilla de inicialización de Amazon EC2 en los grupos de escalado automático.

Operaciones de la API compatibles con plantillas de lanzamiento

Para obtener una lista de las operaciones de API compatibles con las plantillas de lanzamiento, consulte Acciones de Amazon EC2 en la Referencia de API de Amazon EC2.

Creación de una plantilla de lanzamiento para un grupo de Auto Scaling

Para poder crear un grupo de escalado automático utilizando una plantilla de lanzamiento, debe crear una plantilla que contenga la información de configuración necesaria para lanzar una instancia, incluido el ID de la Imagen de máquina de Amazon (AMI).

Utilice el siguiente procedimiento para crear nuevas plantillas de lanzamiento.

Contenido

- Creación de una plantilla de lanzamiento (consola)
- Cambo de la configuración de la interfaz de red predeterminada (consola)
- Modificación de la configuración de almacenamiento (consola)
- Creación de una plantilla de lanzamiento a partir de una instancia existente (consola)
- Recursos relacionados
- Limitaciones

> **Important**
>
> Los parámetros de la plantilla de lanzamiento no están completamente validados cuando crea dicha plantilla. Si especifica valores incorrectos para parámetros, o si no utiliza combinaciones de parámetros compatibles, no se puede iniciar ninguna instancia con esta plantilla de lanzamiento. Asegúrese de especificar los valores de parámetros correctos y utilice las combinaciones de parámetros admitidas. Por ejemplo, para lanzar instancias con una AMI de Arm-based AWS Graviton o Graviton2, debe especificar Arm-compatible un tipo de instancia. Para obtener más información, consulte Restricciones de las plantillas de lanzamiento en la Guía del usuario de Amazon EC2.

Creación de una plantilla de lanzamiento (consola)

Los siguientes pasos describen cómo configurar una plantilla de lanzamiento básica:

- Especifique la imagen de máquina de Amazon (AMI) desde la que desea lanzar las instancias.
- Elija un tipo de instancia que sea compatible con la AMI que especifique.

- Especifique el par de claves que se usará al conectarse a instancias, por ejemplo, mediante SSH.
- Agregue uno o varios grupos de seguridad para permitir el acceso a las instancias.
- Especifique si desea adjuntar volúmenes adicionales a cada instancia.
- Agregue etiquetas personalizadas (pares clave-valor) a las instancias y volúmenes.

Para crear una plantilla de lanzamiento

1. Abra la consola de Amazon EC2 en https://console.aws.amazon.com/ec2/.
2. En el panel de navegación, en Instances, seleccione Launch Templates.
3. Elija Crear plantilla de inicialización. Escriba un nombre y una descripción para la versión inicial de la plantilla de lanzamiento.
4. (Opcional) En Orientación sobre Auto Scaling, active la casilla de verificación para que Amazon EC2 proporcione orientación que le ayude a crear una plantilla para usarla con Amazon EC2 Auto Scaling.
5. En Launch template contents (Contenido de la plantilla de lanzamiento), rellene todos los campos obligatorios y los campos opcionales según sea necesario.

   a. Application and OS Images (Amazon Machine Image) (Imágenes de aplicaciones y SO [imagen de máquina de Amazon]): (obligatorio) elija el ID de la AMI para las instancias. Puede buscar en todas las AMI disponibles o seleccionar una AMI en la lista Recents (Recientes) o Quick Start (Inicio rápido). Si no ve la AMI que necesita, elija Browse more AMIs (Buscar más AMI) para navegar por el catálogo completo de AMI.

      Para elegir una AMI personalizada, primero debe crear una AMI a partir de una instancia personalizada. Para obtener más información, consulte Creación de una EBS-backed AMI de Amazon en la Guía del usuario de Amazon EC2.

   b. En Instance type (Tipo de instancia), elija un único tipo de instancia que sea compatible con la AMI que ha especificado.

      De manera alternativa, para utilizar la selección del tipo de instancia basada en atributos, elija Avanzada, Especificar atributos del tipo de instancia y luego especifique las siguientes opciones:

      - Number of vCPUs (Número de vCPU): ingrese el número mínimo y máximo de vCPU. Para indicar que no hay límites, ingrese un mínimo de 0 y deje el máximo en blanco.

- Amount of memory (MiB) (Cantidad de memoria [MiB]): ingrese la cantidad mínima y máxima de memoria, en MiB. Para indicar que no hay límites, ingrese un mínimo de 0 y deje el máximo en blanco.
- Expanda Optional instance type attributes (Atributos de tipo de instancia opcionales) y elija Add attribute (Agregar atributo) para limitar aún más los tipos de instancias que se pueden utilizar para cumplir la capacidad deseada. Para obtener información sobre cada atributo, consulte la InstanceRequirementsRequestreferencia de la API de Amazon EC2.
- Resulting instance types (Tipos de instancias resultantes): puede consultar los tipos de instancia que coinciden con los requisitos de computación especificados, como vCPU, memoria y almacenamiento.
- Para excluir tipos de instancias, elija Add attribute (Agregar atributo). Desde la lista de Attribute (Atributo), elija Excluded instance types (Tipos de instancias excluidos). De la lista Attribute Value (Valor de atributo), seleccione los tipos de instancia que desea excluir.

c. Key pair (login) (Par de claves [inicio de sesión]): para Key pair name (Nombre de par de claves), seleccione un par de claves existente o elija Create new key pair (Crear nuevo par de claves) cree uno nuevo. Para obtener más información, consulte Amazon EC2 key pairs and Linux instances en la Guía del usuario de Amazon EC2.

d. Network settings (Configuración de red): para Firewall (security groups) (Firewall [grupos de seguridad]), utilice uno o más grupos de seguridad o deje este campo en blanco y configure uno o más grupos de seguridad como parte de la interfaz de red. Para obtener más información, consulte Grupos de seguridad de Amazon EC2 para instancias de Linux en la Guía del usuario de Amazon EC2.

   Si no especifica ningún grupo de seguridad en la plantilla de lanzamiento, Amazon EC2 utiliza el grupo de seguridad predeterminado para la VPC en la que se lanzará el grupo de Auto Scaling. De forma predeterminada, este grupo de seguridad no permite el tráfico entrante de redes externas. Para obtener más información, consulte Grupos de seguridad predeterminados de las VPC en la Guía del usuario de Amazon VPC.

e. Realice una de las siguientes acciones:

   - Cambie la configuración de la interfaz de red predeterminada. Por ejemplo, puede habilitar o desactivar la característica de direccionamiento IPv4 público, que anula la configuración de asignación automática de direcciones IPv4 públicas en la subred. Para obtener más información, consulte Cambo de la configuración de la interfaz de red predeterminada (consola).

- Omita este paso si desea mantener la configuración predeterminada de la interfaz de red.

f. Realice una de las siguientes acciones:

- Modifique la configuración de almacenamiento. Para obtener más información, consulte Modificación de la configuración de almacenamiento (consola).
- Omita este paso si desea mantener la configuración de almacenamiento predeterminada.

g. En Resource tags (Etiquetas de recursos), proporcione las combinaciones de clave y valor para especificar las etiquetas. Si especifica etiquetas de instancia en la plantilla de lanzamiento y elige propagar las etiquetas del grupo de Auto Scaling a sus instancias, todas las etiquetas se fusionarán. Si se especifica la misma clave de etiqueta para una etiqueta en su plantilla de lanzamiento y una etiqueta en su grupo de Auto Scaling, entonces el valor de la etiqueta del grupo tiene prioridad.

6. (Opcional) Configure las opciones avanzadas. Por ejemplo, puede elegir un rol de IAM que su aplicación pueda utilizar al acceder a otros AWS o especifique los datos de usuario de instancia que se pueden utilizar para realizar tareas de configuración automatizadas comunes después de que se lance una instancia. Para obtener más información, consulte Creación de una plantilla de lanzamiento mediante la configuración avanzada.
7. Cuando esté listo para crear la plantilla de lanzamiento, elija Create launch template (Crear plantilla de lanzamiento).
8. Para crear un grupo de Auto Scaling), elija Create an Auto Scaling group (Crear un grupo de Auto Scaling) en la página de confirmación.

Cambo de la configuración de la interfaz de red predeterminada (consola)

Las interfaces de red proporcionan conectividad a otros recursos de la VPC e Internet. Para obtener más información, consulte Proporcionar conectividad de red para sus instancias de Auto Scaling mediante Amazon VPC.

En esta sección le indicamos cómo cambiar la configuración predeterminada de la interfaz de red. Por ejemplo, puede definir si desea asignar una dirección IPv4 pública a cada instancia en lugar de establecer de forma predeterminada la configuración de asignación automática de direcciones IPv4 públicas en la subred.

Consideraciones y limitaciones

Al modificar la configuración predeterminada de la interfaz de red, tenga en cuenta las siguientes consideraciones y limitaciones:

- Debe configurar los grupos de seguridad como parte de la interfaz de red, no en la sección Security groups (Grupos de seguridad) de la plantilla. No pueden especificar grupos de seguridad en ambos lugares.
- Si especifica algún identificador de interfaz de red existente, solo podrá lanzar una instancia. Esto se debe a que una interfaz de red solo se puede conectar a una instancia a la vez. Para ello, debe usar el AWS CLI o un SDK para crear el grupo Auto Scaling. Al crear el grupo, especifique una única zona de disponibilidad, pero no un ID de subred.

  Debe incluir una interfaz de red existente con un índice de dispositivo de 0. Si lo desea, puede agregar interfaces de red secundarias existentes. La cantidad máxima de interfaces de red depende del tipo de instancia. Todas las interfaces de red especificadas deben estar en la zona de disponibilidad que configure para el grupo Auto Scaling. Las interfaces pueden estar en subredes diferentes, siempre que cada subred esté en esa zona de disponibilidad.
- No puede asignar automáticamente una dirección IPv4 pública si especifica más de una interfaz de red. Tampoco puede especificar índices de dispositivos duplicados en las interfaces de red. Las interfaces de red primaria y secundaria residen en la misma subred.
- Cuando se lanza una instancia, se asigna automáticamente una dirección privada a cada interfaz de red. La dirección proviene del intervalo de CIDR de la subred en la que se lanza la instancia. Para obtener información sobre cómo especificar bloques de CIDR (o intervalos de direcciones IP) para su VPC o subred, consulte la Guía del usuario de Amazon VPC.

Para cambiar la configuración predeterminada de la interfaz de red

1. En Network settings (Configuración de red), amplíe Advanced network configuration (Configuración de red avanzada).
2. Elija Add network interface (Agregar interfaz de red) para configurar la interfaz de red principal y preste atención a los siguientes campos:

   a. Device index (Índice de dispositivos): deje el valor predeterminado, 0, para aplicar los cambios a la interfaz de red principal (eth0).

   b. Network interface (Interfaz de red): conserve el valor predeterminado, New interface (Nueva interfaz), para que Amazon EC2 Auto Scaling cree automáticamente una nueva interfaz de red cuando se lance una instancia. Como alternativa, puede elegir una interfaz de red existente y disponible con un índice de dispositivos de 0, pero esto limita el grupo de Auto Scaling a una instancia.

   c. Description (Descripción): (opcional) escriba un nombre descriptivo.

d. Subnet (Subred): conserve la configuración predeterminada Don't include in launch template (No incluir en la plantilla de lanzamiento).

   Si la AMI especifica una subred para la interfaz de red, se produce un error. Recomendamos desactivar Auto Scaling guidance (Guía de Auto Scaling) como solución temporal. Después de hacer este cambio, no recibirá ningún mensaje de error. Sin embargo, independientemente de dónde se especifique la subred, la configuración de subred del grupo de Auto Scaling tiene prioridad y no se puede anular.

e. Auto-assign IP pública: cambie si la interfaz de red con un índice de dispositivo igual a 0 recibe una dirección IPv4 pública. De forma predeterminada, las instancias en una subred predeterminada reciben una dirección IPv4 pública, mientras que las instancias en una subred no predeterminada no la reciben. Seleccione Enable (Habilitar) o Disable (Deshabilitar) para anular la configuración predeterminada de la subred.

f. Security groups (Grupos de seguridad): elija uno o varios grupos de seguridad para la interfaz de red. Cada grupo de seguridad debe configurarse para la VPC en la que el grupo de Auto Scaling lanzará las instancias. Para obtener más información, consulte Grupos de seguridad de Amazon EC2 para instancias de Linux en la Guía del usuario de Amazon EC2.

g. Delete on termination (Eliminar al terminar): seleccione Yes (Sí) para eliminar la interfaz de red cuando se termina la instancia o elija No si desea conservarla.

h. Elastic Fabric Adapter: para admitir casos de uso de computación de alto rendimiento y machine learning, cambie la interfaz de red a una interfaz de red de Elastic Fabric Adapter. Para obtener más información, consulte Elastic Fabric Adapter en la Guía del usuario de Amazon EC2.

i. Network card index (Índice de tarjetas de red): seleccione 0 para adjuntar la interfaz de red principal a la tarjeta de red con un índice de dispositivos de 0. Si esta opción no está disponible, deje el valor predeterminado, Don't include in launch template (No incluir en la plantilla de lanzamiento). La conexión de la interfaz de red a una tarjeta de red específica está disponible solo para los tipos de instancias compatibles. Para obtener más información, consulte Tarjetas de red en la Guía del usuario de Amazon EC2.

j. ENA Express: para tipos de instancias compatibles con ENA Express, puede seleccionar Habilitar para habilitar ENA Express o Deshabilitar para deshabilitarlo. Para obtener más información, consulte Improve network performance with ENA Express on Linux instances en la Guía del usuario de Amazon EC2.

k. UDP de ENA Express: si habilita ENA Express, puede usarla opcionalmente para el tráfico UDP. Seleccione Habilitar para activar el UDP de ENA Express o Deshabilitar para deshabilitarlo.

3. Para agregar una interfaz de red secundaria, elija Add network interface (Agregar interfaz de red).

Modificación de la configuración de almacenamiento (consola)

Puede modificar la configuración de almacenamiento de las instancias lanzadas desde una AMI de Amazon o una EBS-backed AMI con respaldo en un almacén de instancias. También puede especificar volúmenes de EBS adicionales para adjuntar a la instancias. La AMI incluye uno o más volúmenes de almacenamiento, incluido el volumen raíz (Volumen 1 [raíz de AMI]).

Para modificar la configuración de almacenamiento

1. En Configure storage (Configurar almacenamiento), modifique el tamaño o el tipo de volumen.

   Si el valor especificado para el tamaño del volumen supera los límites del tipo de volumen o es inferior al tamaño de la instantánea, se muestra un mensaje de error. Para ayudarlo a solucionar el problema, este mensaje proporciona el valor mínimo o máximo que el campo puede aceptar.

   Solo aparecen los volúmenes asociados a una Amazon EBS-backed AMI. Para mostrar información sobre la configuración de almacenamiento de una instancia lanzada desde una AMI con almacenamiento de instancias, elija Show details (Mostrar detalles) en la sección Instance store volumes (Volúmenes de almacén de instancias).

   Si desea especificar todos los parámetros de volumen de EBS, cambie a la vista Advanced (Avanzada) en la esquina superior derecha.

2. Para obtener opciones avanzadas, amplíe el volumen que desea modificar y configure el volumen de la siguiente manera:

   a. Storage type (Tipo de almacenamiento): el tipo de volumen (EBS o efímero) que desea asociar a la instancia. El tipo de volumen del almacén de instancias (efímero) solo está disponible si selecciona un tipo de instancia que lo admita. Para obtener más información, consulte Volúmenes de Amazon EBS en la Guía del usuario de Amazon EBS y Almacén de instancias de Amazon EC2 en la Guía del usuario de Amazon EC2.

b. Device name (Nombre de dispositivo): selecciónelo de la lista de nombres de dispositivo disponibles para el volumen.

c. Snapshot (Instantánea): seleccione la instantánea desde la que desea crear un volumen. También puede buscar instantáneas públicas y compartidas disponibles escribiendo texto en el campo Snapshot (Instantánea).

d. Size (GiB) (Tamaño [GiB]): para volúmenes de EBS, puede especificar un tamaño de almacenamiento. Si ha seleccionado una AMI y una instancia aptas para el nivel gratuito, recuerde que para permanecer en dicho nivel debe mantenerse por debajo de los 30 GiB de almacenamiento total. Para obtener más información, consulte Restricciones de tamaño y configuración de un volumen de EBS en la Guía del usuario de Amazon EBS.

e. Volume type (Tipo de volumen): elija un tipo de volumen para los volúmenes de EBS. Para obtener más información, consulte Tipos de volúmenes de Amazon EBS en la Guía del usuario de Amazon EBS.

f. IOPS: si ha seleccionado un tipo de volumen de SSD (`io1`y`io2`) o SSD de uso general (`gp3`) con IOPS aprovisionadas, puede introducir el número de I/O operaciones por segundo (IOPS) que admite el volumen. Es necesario para los volúmenes io1, io2 y gp3. No se admite para los volúmenes gp2, st1, sc1 o estándar.

g. Delete on termination (Eliminar al terminar): para los volúmenes de EBS, elija Yes (Sí) para eliminar el volumen cuando se termine la instancia asociada o elija No para conservarlo.

h. Encrypted (Cifrado): si el tipo de instancia admite el cifrado EBS, puede elegir Yes (Sí) para habilitar el cifrado para el volumen. Si ha habilitado el cifrado de forma predeterminada en esta región, el cifrado se habilita automáticamente. Para obtener más información, consulte Cifrado de Amazon EBS y Habilitación del cifrado de Amazon EBS de manera predeterminada en la Guía del usuario de Amazon EBS.

   El efecto predeterminado de configurar este parámetro varía según la elección de volumen de origen, tal y como se describe en la tabla que se muestra a continuación. En todos los casos, debe tener permiso para usar el especificado. AWS KMS key

Resultados del cifrado

Si el parámetro **`Encrypted`** se establece en…	Y si el origen del volumen es…	El estado predeterminado del cifrado es…	Notas
No	Nuevo volumen (vacío)	Sin cifrar*	N/A
	Instantánea no cifrada que posea	Sin cifrar*	
	Instantánea cifrada que posea	Cifrada con la misma clave	
	Instantánea no cifrada compartida con usted	Sin cifrar*	
	Instantánea cifrada compartida con usted	Cifrado con la clave de KMS predeterminada	
Sí	Nuevo volumen	Cifrado con la clave de KMS predeterminada	Para utilizar una clave KMS que no sea la predeterminada, especifique un valor para el parámetro de KMS Key (Clave KMS).
	Instantánea no cifrada que posea	Cifrado con la clave de KMS predeterminada	
	Instantánea cifrada que posea	Cifrada con la misma clave	
	Instantánea no cifrada compartida con usted	Cifrado con la clave de KMS predeterminada	

Si el parámetro **`Encrypted`** se establece en…	Y si el origen del volumen es…	El estado predeterminado del cifrado es…	Notas
	Instantánea cifrada compartida con usted	Cifrado con la clave de KMS predeterminada	

* Si el cifrado predeterminado está habilitado, todos los volúmenes que acaba de crear (tanto si el parámetro de Encrypted [Cifrado] está establecido en Yes [Sí] como si no lo está) se cifran mediante la clave KMS predeterminada. Si establece los parámetros Encrypted (Cifrado) y Key (Clave), entonces puede especificar una clave KMS no predeterminada.

i. KMS key (Clave KMS): si ha elegido Yes (Sí) para Encrypted (Cifrado), a continuación, debe seleccionar una clave administrada por el cliente a fin de utilizarla para cifrar el volumen. Si ha habilitado el cifrado de forma predeterminada en esta región, se selecciona automáticamente la clave predeterminada administrada por el cliente. Puede seleccionar una clave diferente o especificar el ARN de cualquier clave administrada por el cliente que haya creado anteriormente con AWS Key Management Service.

3. A fin de especificar volúmenes adicionales para adjuntar a las instancias lanzadas por esta plantilla de lanzamiento, elija Add new volume (Agregar nuevo volumen).

Creación de una plantilla de lanzamiento a partir de una instancia existente (consola)

Para crear una plantilla de lanzamiento a partir de una instancia disponible

1. Abra la consola de Amazon EC2 en https://console.aws.amazon.com/ec2/.
2. En el panel de navegación, en Instances (Instancias), elija Instances.
3. Seleccione la instancia y elija Actions (Acciones), Image and templates (Imagen y plantillas), Create Template from Instance (Crear plantilla desde instancia).
4. Proporcione un nombre y una descripción.
5. En Auto Scaling guidance (Guía de Auto Scaling), seleccione la casilla de verificación.

6. Ajuste la configuración según sea necesario y elija Create launch template (Crear plantilla de lanzamiento).
7. Para crear un grupo de Auto Scaling, elija Create an Auto Scaling group (Crear un grupo de Auto Scaling) en la página de confirmación.

Recursos relacionados

Te proporcionamos algunos fragmentos de plantillas de JSON y YAML que puedes usar para entender cómo declarar las plantillas de lanzamiento en tus CloudFormation plantillas de stack. Para obtener más información, consulte las secciones AWS: :EC2:: LaunchTemplate y Creación de plantillas de lanzamiento con CloudFormation las secciones de la Guía del AWS CloudFormation usuario.

Para obtener más información sobre las plantillas de lanzamiento, consulte Lanzamiento de una instancia desde una plantilla de lanzamiento en la Guía del usuario de Amazon EC2.

Limitaciones

- Si bien puede especificar una subred en una plantilla de lanzamiento, no es necesario hacerlo si solo usa la plantilla de lanzamiento para crear grupos de escalado automático. No puede especificar la subred de un grupo de escalado automático especificando la subred en una plantilla de lanzamiento. Las subredes del grupo de escalado automático se toman de la propia definición de recursos del grupo de escalado automático.
- Para ver otras limitaciones en las interfaces de red definidas por el usuario, consulte Cambo de la configuración de la interfaz de red predeterminada (consola).

Creación de una plantilla de lanzamiento mediante la configuración avanzada

En este tema, se describe cómo crear una plantilla de lanzamiento con la configuración avanzada de la Consola de administración de AWS.

Creación de una plantilla de inicialización mediante la configuración avanzada

1. Abra la consola de Amazon EC2 en https://console.aws.amazon.com/ec2/.

2. En el panel de navegación, en Instancias elija Plantillas de inicialización y, a continuación, Crear plantilla de inicialización.
3. Configure la plantilla de inicialización como se describe en los siguientes temas:
   - Configuración necesaria
   - Configuración avanzada
4. Elija Crear plantilla de inicialización.

Configuración necesaria

Al momento de crear una plantilla de inicialización, debe incluir los siguientes ajustes necesarios.

Nombre de plantilla de inicialización

Ingrese un nombre único que describa la plantilla de inicialización.

Imágenes de aplicaciones y sistema operativo (Imagen de máquina de Amazon)

Elija la imagen de máquina de Amazon (AMI) que desea utilizar. Puede buscar la AMI que desea utilizar. Para lograr una mayor eficiencia de escalado, elija una AMI personalizada que esté completamente configurada para inicializar una instancia con el código de la aplicación y que requiera pocas modificaciones durante la inicialización.

Tipo de instancia

Elija un tipo de instancia que sea compatible con la AMI. Puede omitir la adición de un tipo de instancia a su plantilla de inicialización si planea usar varios tipos de instancias que estén integrados en la propia definición de recursos del grupo de escalado automático. El tipo de instancia solo es necesario si no tiene pensado crear un grupo de instancias mixto.

Configuración avanzada

La configuración avanzada es opcional. Si no establece ninguna configuración avanzada, las capacidades específicas no se agregarán a las instancias.

Expanda la sección Detalles avanzados para ver la configuración avanzada. En las siguientes secciones, se describen las configuraciones avanzadas más útiles a las que se debe prestar atención al momento de crear una plantilla de inicialización para un grupo de escalado automático. Para obtener más información, consulte Detalles avanzados en la Guía del usuario de Amazon EC2.

Perfil de instancia IAM

El perfil de instancia contiene el rol de IAM que desea utilizar. Cuando su grupo de escalado automático inicializa una instancia de EC2, los permisos definidos en el rol de IAM asociado se otorgan a las aplicaciones que se ejecutan en la instancia. Para obtener más información, consulte Rol de IAM para aplicaciones que se ejecuten en instancias de Amazon EC2.

Protección de finalización

Cuando está habilitada, esta característica evita que los usuarios finalicen una instancia utilizando la consola de Amazon EC2, los comandos de la CLI y las operaciones de la API. La protección contra finalización proporciona una protección adicional contra la finalización accidental. Esto no impide que Amazon EC2 Auto Scaling finalice una instancia. Para controlar qué instancias se pueden finalizar con Amazon EC2 Auto Scaling, consulte Uso de la protección frente a la reducción horizontal de instancias para controlar la terminación de instancias.

Supervisión detallada CloudWatch

Puede habilitar la supervisión detallada de sus instancias EC2 para que puedan enviar datos de métricas a Amazon CloudWatch en intervalos de 1 minuto. De forma predeterminada, las instancias EC2 envían los datos métricos a CloudWatch intervalos de 5 minutos. Se aplican cargos adicionales de . Para obtener más información, consulte Configuración de la supervisión para instancias de Auto Scaling.

Especificación de créditos

Amazon EC2 proporciona instancias de rendimiento ampliable, como T2, T3 y T3a, que permiten a las aplicaciones superar el rendimiento básico de la CPU cuando es necesario. De forma predeterminada, estas instancias pueden reproducirse durante un tiempo limitado antes de que se limite el uso de la CPU. De manera opcional, puede habilitar el modo ilimitado para que las instancias puedan superar la base de referencia por el tiempo que sea necesario. Esto permite que las aplicaciones mantengan un alto rendimiento de la CPU cuando sea necesario. Es posible que se apliquen cargos adicionales. Para obtener más información, consulte Uso de un grupo de escalado automático para lanzar una instancia de rendimiento ampliable como ilimitada en la Guía del usuario de Amazon EC2.

Nombre del grupo de ubicación

Puede especificar un grupo de ubicación y utilizar una estrategia de clúster o partición para influir en la ubicación física de las instancias en el centro de datos de AWS . Para grupos pequeños de escalado automático, también puede usar la estrategia de dispersión. Para obtener más información, consulte Grupos de ubicación en la Guía del usuario de Amazon EC2.

Hay algunas consideraciones que se deben tener en cuenta al momento de usar grupos de ubicación con grupos de escalado automático:

- Si se especifica un grupo de ubicación en la plantilla de inicialización y en el grupo de escalado automático, el grupo de ubicación del grupo de escalado automático tiene prioridad.
- En CloudFormation, tenga cuidado al definir un grupo de ubicaciones en la plantilla de lanzamiento. Amazon EC2 Auto Scaling inicializará las instancias en el grupo de ubicación especificado. Sin embargo, no CloudFormation recibirá señales de esas instancias si usa una UpdatePolicycon su grupo de Auto Scaling (aunque esto podría cambiar en el futuro).

Opciones de compra

Puede elegir Solicitar instancias puntuales para solicitar instancias puntuales al precio puntual, con un precio limitado, y elegir Personalizar para cambiar la configuración predeterminada de las instancias puntuales. On-Demand Para un grupo de Auto Scaling, debe especificar una solicitud puntual sin fecha de finalización (la predeterminada). Para obtener más información, consulte Solicitud de instancias de spot para aplicaciones flexibles y tolerantes a errores. Esta configuración puede resultar útil en circunstancias especiales, pero en general es mejor no especificarla y, en su lugar, crear un grupo de instancias mixtas. Para obtener más información, consulte Grupos de Auto Scaling con varios tipos de instancia y opciones de compra.

Si especifica una solicitud de instancia de spot en su plantilla de lanzamiento, no podrá crear un grupo de instancias mixtas. Si intenta utilizar una plantilla de lanzamiento que solicite instancias de spot con un grupo de instancias mixtas, recibirá el siguiente mensaje de error: `Incompatible launch template: You cannot use a launch template that is set to request Spot Instances (InstanceMarketOptions) when you configure an Auto Scaling group with a mixed instances policy. Add a different launch template to the group and try again.`

Capacity Reservation

Las reservas de capacidad le permiten reservar capacidad para sus instancias de Amazon EC2 en una zona de disponibilidad específica por cualquier período. Para obtener más información, consulte Reservas On-Demand de capacidad en la Guía del usuario de Amazon EC2.

Puede elegir si quiere inicializar instancias en:

- cualquier reserva de capacidad abierta (Abierta)
- una reserva de capacidad específica (Objetivo por ID)
- un grupo de reservas de capacidad (Objetivo por grupo)

Para segmentar una reserva de capacidad específica, el tipo de instancia de la plantilla de inicialización debe coincidir con el tipo de instancia de la reserva. Cuando cree su grupo de escalado automático, use la misma zona de disponibilidad que la reserva de capacidad. En función de la Región de AWS que elija, puede optar por centrarse en un bloque de capacidad. Para obtener más información, consulte Uso Bloques de capacidad para cargas de trabajo de aprendizaje automático.

Para dirigirse a un grupo de reservas de capacidad, consulte Capacidad reservada en zonas de disponibilidad específicas con reserva de capacidad. Si se dirige a un grupo de reservas de capacidad, puede distribuir la capacidad en varias zonas de disponibilidad para mejorar la resiliencia.

Propiedad

Amazon EC2 ofrece tres opciones para la tenencia de las instancias de EC2:

- Compartida (Compartida): varias Cuentas de AWS pueden compartir el mismo hardware físico. Esta es la opción de tenencia predeterminada cuando se inicializa una instancia.
- Instancia dedicada (Dedicada): la instancia se ejecuta en el hardware de un solo inquilino. Ningún otro cliente de AWS comparte el mismo servidor físico. Para obtener más información, consulte Instancias dedicadas en la Guía del usuario de Amazon EC2.
- Hosts dedicados (Host dedicado): la instancia se ejecuta en un servidor físico dedicado para su uso. El uso de hosts dedicados hace que sea más fácil traer su propia licencia (BYOL) con requisitos de hardware específicos a EC2 y cumplir con los casos de uso. Si elige esta opción, debe proporcionar un grupo de recursos de host para el grupo de recursos de host de tenencia. Para obtener más información, consulte Hosts dedicados en la Guía del usuario de Amazon EC2.

La compatibilidad con hosts dedicados solo está disponible si especifica un grupo de recursos de host. No puede dirigirse a un ID de host específico ni utilizar la afinidad de ubicación de host.

- Si intenta utilizar una plantilla de inicialización que especifique un ID de host, recibirá el siguiente mensaje de error: `Incompatible launch template: Tenancy host ID is not supported for Auto Scaling.`
- Si intenta utilizar una plantilla de inicialización que especifique la afinidad de ubicación de host, recibirá el siguiente mensaje de error: `Incompatible launch template: Auto Scaling does not support host placement affinity.`

Grupo de recursos para host de tenencia

Con AWS License Managerél, puede incorporar sus propias licencias AWS y administrarlas de forma centralizada. Un grupo de recursos de host es un grupo de hosts dedicados que están vinculados a una configuración de licencias de License Manager específica. Los grupos de recursos de host le permiten lanzar fácilmente instancias de EC2 en hosts dedicados que se adapten a sus necesidades de licencias de software. No es necesario asignar de forma manual los hosts dedicados con antelación. Se crean automáticamente según sea necesario. Tenga en cuenta que si asocia una AMI a una configuración de licencias, esa AMI solo se puede asociar a un grupo de recursos de host a la vez. Para obtener más información, consulte Grupos de recursos de host en AWS License Manager en la Guía del usuario de License Manager.

Configuraciones de licencias

Con esta configuración, puede especificar una configuración de licencia para sus instancias sin restringir su tenencia a los hosts dedicados. La configuración de licencias hace un seguimiento de las licencias de software implementadas en las instancias para que pueda supervisar el uso y el cumplimiento de las licencias. Para obtener más información, consulte la sección Cree una licencia autoadministrada en la Guía del usuario de License Manager.

Metadatos accesibles

Puede elegir si habilitar o deshabilitar el acceso al punto de conexión HTTP del servicio de metadatos de la instancia. De forma predeterminada, el punto de conexión HTTP está habilitado. Si decide desactivar el punto de conexión, el acceso a los metadatos de la instancia está desactivado. Puede especificar la condición para requerir IMDSv2 solo cuando el punto de conexión HTTP está habilitado. Para obtener más información, consulte Configurar las opciones de metadatos de instancia en la Guía del usuario de Amazon EC2.

Versión de metadatos

Puede optar por requerir el uso de Servicio de metadatos de instancia versión 2 (IMDSv2) al solicitar metadatos de instancia. Si no especifica un valor, el valor predeterminado es admitir IMDSV1 e IMDSv2. Para obtener más información, consulte Configurar las opciones de metadatos de instancia en la Guía del usuario de Amazon EC2.

Límite de saltos de respuesta del token de metadatos

Puede establecer el número permitido de saltos de red para el token de metadatos. Si no especifica un valor, el predeterminado es 1. Para obtener más información, consulte Configurar las opciones de metadatos de instancia en la Guía del usuario de Amazon EC2.

Datos de usuario

Puede personalizar y terminar de configurar las instancias en el momento del inicio especificando los scripts del intérprete de comandos o las directivas cloud-init como datos de usuario. Los datos de usuario se ejecutan cuando la instancia se inicia por primera vez, lo que le permite instalar aplicaciones, dependencias o personalizaciones automáticamente en el momento del inicio. Para obtener más información, consulte Ejecutar comandos en la instancia de Linux durante la inicialización en la Guía del usuario de Amazon EC2.

Si tiene grandes descargas o scripts complejos, esto aumenta el tiempo que tarda la instancia en estar lista para su uso. En ese caso, es posible que tenga que configurar un enlace de ciclo de vida para retrasar que una instancia alcance el estado `InService` hasta que esté completamente aprovisionada. Para obtener más información acerca de cómo agregar un enlace de ciclo de vida a su grupo de escalado automático, consulte Enlaces de ciclo de vida de Amazon EC2 Auto Scaling.

Solicitud de instancias de spot para aplicaciones flexibles y tolerantes a errores

En la plantilla de lanzamiento, puede solicitar de manera opcional instancias de spot sin fecha de finalización ni duración. Las instancias puntuales Amazon EC2 son capacidad sobrante y están disponibles con grandes descuentos en comparación con el precio de On-Demand EC2. Las instancias de spot son una opción económica si es flexible con respecto a cuándo es necesario ejecutar las aplicaciones y si las aplicaciones se pueden interrumpir. Para obtener más información sobre cómo crear una plantilla de lanzamiento que solicite instancias de spot, consulte Creación de una plantilla de lanzamiento mediante la configuración avanzada.

Important

Las instancias puntuales se utilizan normalmente para complementar On-Demand las instancias. Para ese caso, puede especificar la misma configuración que se utiliza para lanzar instancias de spot durante la configuración de un grupo de Auto Scaling. Al especificar la configuración como parte del grupo Auto Scaling, puede solicitar el lanzamiento de instancias puntuales solo después de lanzar un número determinado de On-Demand instancias y, a continuación, continuar lanzando alguna combinación de On-Demand instancias e instancias puntuales a medida que el grupo vaya escalando. Para obtener más

información, consulte Grupos de Auto Scaling con varios tipos de instancia y opciones de compra.

En este tema se describe cómo se lanzan únicamente instancias de spot en un grupo de Auto Scaling mediante la especificación de la configuración en una plantilla de lanzamiento, en lugar de hacerlo en el propio grupo de Auto Scaling. La información de este tema también se aplica a los grupos de Auto Scaling que solicitan instancias de spot con una configuración de lanzamiento. La diferencia radica en que una configuración de lanzamiento requiere un precio máximo, pero para las plantillas de lanzamiento, el precio máximo es opcional.

Cuando cree una plantilla de lanzamiento para lanzar solo instancias de spot, tenga en cuenta los siguientes aspectos:

- Precio de spot. Solo pagará el precio de spot actual por las instancias de spot que lance. Estos precios cambian gradualmente con el paso del tiempo en función de las tendencias a largo plazo de la oferta y la demanda. Para obtener más información, consulte Instancias de spot y Precios y ahorros en la Guía del usuario de Amazon EC2.
- Definición del precio máximo. Si lo desea, puede incluir un precio máximo por hora para las instancias de spot en su plantilla de lanzamiento. Si el precio máximo es superior al precio de spot actual, el servicio de spot de Amazon EC2 atiende inmediatamente su solicitud si hay capacidad disponible. Si el precio de las instancias de spot supera el precio máximo para una instancia de ejecución en su grupo de Auto Scaling, termina la instancia.

  > **Warning**
  >
  > Su aplicación podría no ejecutarse si no recibe sus instancias de spot, como cuando el precio máximo es demasiado bajo. Para aprovechar las instancias puntuales disponibles durante el mayor tiempo posible, fije el precio máximo cercano al On-Demand precio.

- Equilibrio entre zonas de disponibilidad. Si especifica varias zonas de disponibilidad, Amazon EC2 Auto Scaling distribuye las solicitudes de spot entre las zonas especificadas. Si el precio máximo es demasiado bajo en una zona de disponibilidad para que se atiendan las solicitudes, Amazon EC2 Auto Scaling comprueba si se han atendido solicitudes en otras zonas. En tal caso, Amazon EC2 Auto Scaling cancela las solicitudes que no se han atendido y las redistribuye entre las zonas de disponibilidad que tienen solicitudes atendidas. Si el precio en una zona de disponibilidad sin solicitudes atendidas se reduce lo suficiente como para poder atender solicitudes futuras, Amazon EC2 Auto Scaling reequilibra todas las zonas de disponibilidad.

- Terminación de instancias de spot. Las instancias de spot pueden terminarse en cualquier momento. El servicio de spot de Amazon EC2 puede terminar las instancias de spot en su grupo de Auto Scaling a medida que cambie la disponibilidad o el precio de estas. Al escalar o realizar comprobaciones de estado, Amazon EC2 Auto Scaling también puede terminar las instancias puntuales del mismo modo que puede On-Demand terminar las instancias. Cuando se termina una instancia, se elimina cualquier tipo de almacenamiento.
- Mantenimiento de la capacidad deseada. Cuando se termina una instancia de spot, Amazon EC2 Auto Scaling intenta lanzar otra instancia de spot para mantener la capacidad deseada del grupo. Si el precio de spot actual es inferior al precio máximo, se lanza una instancia de spot. Si la solicitud de una instancia de spot no se realiza correctamente, lo sigue intentando.
- Cambio del precio máximo. Si desea cambiar su precio máximo, cree una nueva plantilla de lanzamiento o actualice una ya existente con el nuevo precio máximo y, a continuación, asóciela a su grupo de Auto Scaling. Las instancias de spot existentes seguirán ejecutándose siempre y cuando el precio máximo especificado en la plantilla de lanzamiento utilizada para esas instancias sea superior que el precio de spot actual. Si no ha establecido un precio máximo, el precio máximo predeterminado es el On-Demand precio.

Uso Bloques de capacidad para cargas de trabajo de aprendizaje automático

Capacity Blocks lo ayuda a reservar instancias de GPU que tienen una alta demanda para el futuro a fin de respaldar sus cargas de trabajo de machine learning (ML) de corta duración.

Para obtener información general sobre Capacity Blocks y su funcionamiento, consulte Capacity Blocks para ML en la Guía del usuario de Amazon EC2.

Para comenzar a utilizar Capacity Blocks, cree una reserva de capacidad en una zona de disponibilidad específica. Los Capacity Blocks se entregan como reservas de capacidad de `targeted` en una única zona de disponibilidad. Al crear la plantilla de inicialización, especifique el ID de reserva y el tipo de instancia del bloque de capacidad. A continuación, actualice su grupo de escalado automático para usar la plantilla de lanzamiento que creó y la zona de disponibilidad del bloque de capacidad. Cuando comience su reserva de bloque de capacidad, utilice el escalado programado para lanzar la misma cantidad de instancias que su reserva de bloque de capacidad.

> **Important**
>
> Capacity Blockssolo están disponibles para determinados tipos de instancias de Amazon EC2 y. Regiones de AWS Para obtener más información, consulte Requisitos previos en la Guía del usuario de Amazon EC2.

Contenido

- Directrices operativas
- Especificación de un bloque de capacidad en la plantilla de lanzamiento
- Limitaciones
- Recursos relacionados

Directrices operativas

A continuación, se detallan las directrices operativas básicas que se deben seguir al utilizar un bloque de capacidad con un grupo de escalado automático.

- Reduzca horizontalmente su grupo de escalado automático a cero más de 30 minutos antes de la hora de finalización de la reserva del bloque de capacidad. Amazon EC2 terminará todas las instancias que sigan ejecutándose 30 minutos antes de la hora de finalización del bloque de capacidad.
- Le recomendamos que utilice el escalado programado para escalar horizontalmente (añadir instancias) y reducir horizontalmente (eliminar instancias) en los momentos de reserva adecuados. Para obtener más información, consulte Escalado programado para Amazon EC2 Auto Scaling.
- Añada enlaces de ciclo de vida según sea necesario para cerrar correctamente la aplicación dentro de las instancias al reducir horizontalmente. Deje tiempo suficiente para que se complete la acción del ciclo de vida antes de que Amazon EC2 comience a terminar sus instancias por la fuerza 30 minutos antes de la hora de finalización de la reserva de bloques de capacidad. Para obtener más información, consulte Enlaces de ciclo de vida de Amazon EC2 Auto Scaling.
- Asegúrese de que el grupo de escalado automático apunte a la versión correcta de la plantilla de lanzamiento durante toda la reserva. Recomendamos apuntar a una versión específica de la plantilla de lanzamiento en lugar de la versión `$Default` o `$Latest`.

> **Note**
>
> Si deja una instancia de bloque de capacidad en ejecución hasta el final de la reserva y Amazon EC2 la recupera, las actividades de escalado de su grupo de escalado automático indican que era «`taken out of service in response to an EC2 health check that indicated it had been terminated or stopped`», aunque se haya reclamado deliberadamente al final del bloque de capacidad. Del mismo modo, Amazon EC2 Auto Scaling intentará reemplazar la instancia de la misma manera que lo hace con cualquier instancia que no supere una comprobación de estado. Para obtener más información, consulte Comprobaciones de estado para instancias en un grupo de escalado automático.

Especificación de un bloque de capacidad en la plantilla de lanzamiento

Para crear una plantilla de inicialización que se dirija a un bloque de capacidad específico para su grupo de escalado automático, use uno de los siguientes métodos:

Console

Especificación de un bloque de capacidad en la plantilla de lanzamiento (consola)

1. Abra la consola de Amazon EC2 en https://console.aws.amazon.com/ec2/.
2. En la barra de navegación superior, seleccione el Región de AWS lugar donde creó su bloque de capacidad.
3. En el panel de navegación, en Instances, seleccione Launch Templates.
4. Seleccione Crear plantilla de lanzamiento y cree la plantilla de lanzamiento. Incluya el ID de la Imagen de máquina de Amazon (AMI), el tipo de instancia y cualquier otra configuración de plantilla de lanzamiento, según sea necesario.
5. Expanda la sección Detalles avanzados para ver la configuración avanzada.
6. En Opción de compra, elija Bloques de capacidad.
7. En Reserva de capacidad, elija Destino por ID y, a continuación, en Reserva de capacidad - Destino por ID, elija el ID de reserva de capacidad de un bloque de capacidad existente.
8. Cuando haya terminado, seleccione Crear plantilla de lanzamiento.

   Para obtener ayuda para crear un grupo de escalado automático con una plantilla de lanzamiento, consulte Creación de un grupo de Auto Scaling mediante una plantilla de lanzamiento.

AWS CLI

Para especificar un bloque de capacidad en la plantilla de lanzamiento (AWS CLI)

Utilice el siguiente comando create-launch-template para crear una plantilla de lanzamiento que especifique un ID de reserva de bloque de capacidad existente. Reemplace cada *user input placeholder* por su propia información.

```
aws ec2 create-launch-template --launch-template-name my-template-for-capacity-block
  \
  --version-description AutoScalingVersion1 --region us-east-2 \
  --launch-template-data file://config.json
```

> **Tip**
>
> Si este comando arroja un error, asegúrese de haber actualizado la versión AWS CLI local a la última versión.

Contenido de `config.json`.

```
{
    "ImageId": "ami-04d5cc9b88example",
    "InstanceType": "p4d.24xlarge",
    "SecurityGroupIds": [
        "sg-903004f88example"
    ],
    "KeyName": "MyKeyPair",
    "InstanceMarketOptions": {
        "MarketType": "capacity-block"
    },
    "CapacityReservationSpecification": {
        "CapacityReservationTarget": {
            "CapacityReservationId": "cr-02168da1478b509e0"
        }
    }
}
```

A continuación, se muestra un ejemplo del resultado.

```
{
```

```
    "LaunchTemplate": {
        "LaunchTemplateId": "lt-068f72b724example",
        "LaunchTemplateName": "my-template-for-capacity-block",
        "CreateTime": "2023-10-27T15:12:44.000Z",
        "CreatedBy": "arn:aws:iam::123456789012:user/Bob",
        "DefaultVersionNumber": 1,
        "LatestVersionNumber": 1
    }
}
```

Puede usar el siguiente comando describe-launch-template-versions para verificar el ID de reserva del bloque de capacidad asociado a la plantilla de lanzamiento.

```
aws ec2 describe-launch-template-versions --launch-template-names my-template-for-capacity-block \
  --region us-east-2
```

A continuación, se muestra un resultado de ejemplo para una plantilla de lanzamiento que especifica una reserva de bloque de capacidad.

```
{
    "LaunchTemplateVersions": [
        {
            "LaunchTemplateId": "lt-068f72b724example",
            "LaunchTemplateName": "my-template-for-capacity-block",
            "VersionNumber": 1,
            "CreateTime": "2023-10-27T15:12:44.000Z",
            "CreatedBy": "arn:aws:iam::123456789012:user/Bob",
            "DefaultVersion": true,
            "LaunchTemplateData": {
                "ImageId": "ami-04d5cc9b88example",
                "InstanceType": "p5.48xlarge",
                "SecurityGroupIds": [
                    "sg-903004f88example"
                ],
                "KeyName": "MyKeyPair",
                "InstanceMarketOptions": {
                    "MarketType": "capacity-block"
                },
                "CapacityReservationSpecification": {
                    "CapacityReservationTarget": {
                        "CapacityReservationId": "cr-02168da1478b509e0"
```

```
                    }
                }
            }
        }
    ]
}
```

Limitaciones

- La compatibilidad con Capacity Blocks solo está disponible si su grupo de escalado automático tiene una configuración compatible. No se admiten grupos de instancias mixtas ni grupos en caliente.
- Solo puede dirigirse a un bloque de capacidad a la vez.

Recursos relacionados

- Para conocer los requisitos previos y las recomendaciones para el uso de instancias P5, consulte Introducción a las instancias P5 en la Guía del usuario de Amazon EC2.
- Amazon EKS admite el uso de Capacity Blocks para complementar sus cargas de trabajo de machine learning (ML) de corta duración en clústeres de Amazon EKS. Para obtener más información, consulte Capacity Blocks para ML en la Guía del usuario de Amazon EKS.
- Puede usar Capacity Blocks con los tipos de instancias y regiones admitidos. Sin embargo, las reservas de On-Demand capacidad ofrecen flexibilidad para reservar capacidad para otros tipos de instancias y regiones. Para ver un tutorial que muestra cómo utilizar la opción de reserva de On-Demand capacidad, consulteCapacidad reservada en zonas de disponibilidad específicas con reserva de capacidad.

Migración de sus grupos de escalado automático a plantillas de lanzamiento

Desde el 1 de enero de 2023, los nuevos tipos de instancias ya no son compatibles con las configuraciones de lanzamiento. Esto se aplica a cualquier tipo de instancia que se añada y Región de AWS después del lanzamiento inicial de la región. Además, es posible que se pueda crear una configuración de lanzamiento con un tipo de instancia que ya no sea compatible en una región. Para obtener más información, consulte Configuraciones de inicialización de Auto Scaling.

Para migrar un grupo de escalado automático de configuraciones de lanzamiento a plantillas de lanzamiento, consulte los siguientes pasos.

> **Important**
>
> Compruebe que tiene los permisos necesarios para trabajar con las plantillas de lanzamiento. Para obtener más información, consulte Permisos para trabajar con plantillas de lanzamiento. Asegúrese de que los recursos asociados a la configuración de lanzamiento, como los grupos de seguridad y los perfiles de instancia de IAM, no se eliminen involuntariamente. Pruebe la operación de reversión para asegurarse de que los cambios se puedan revertir de forma segura.

Paso 1: buscar grupos de escalado automático que utilicen configuraciones de lanzamiento

Para identificar si tiene grupos de escalado automático que aún utilizan configuraciones de lanzamiento, ejecute el siguiente comando describe-auto-scaling-groups con la AWS CLI. *REGION*Sustitúyala por la tuya Región de AWS.

```
aws autoscaling describe-auto-scaling-groups --region REGION \
  --query 'AutoScalingGroups[?LaunchConfigurationName!=`null`]'
```

A continuación, se muestra un ejemplo del resultado.

```
[
    {
 "AutoScalingGroupName": "group-1",
 "AutoScalingGroupARN": "arn",
 "LaunchConfigurationName": "my-launch-config",
 "MinSize": 1,
 "MaxSize": 5,
 "DesiredCapacity": 2,
 "DefaultCooldown": 300,
 "AvailabilityZones": [
            "us-west-2a",
            "us-west-2b",
            "us-west-2c"
        ],
 "LoadBalancerNames": [],
```

```
"TargetGroupARNs": [],
"HealthCheckType": "EC2",
"HealthCheckGracePeriod": 300,
"Instances": [
          {
              "ProtectedFromScaleIn": false,
              "AvailabilityZone": "us-west-2a",
              "LaunchConfigurationName": "my-launch-config",
              "InstanceId": "i-05b4f7d5be44822a6",
              "InstanceType": "t3.micro",
              "HealthStatus": "Healthy",
              "LifecycleState": "InService"
          },
          {
              "ProtectedFromScaleIn": false,
              "AvailabilityZone": "us-west-2b",
              "LaunchConfigurationName": "my-launch-config",
              "InstanceId": "i-0c20ac468fa3049e8",
              "InstanceType": "t3.micro",
              "HealthStatus": "Healthy",
              "LifecycleState": "InService"
          }
],
"CreatedTime": "2023-03-09T22:15:11.611Z",
"SuspendedProcesses": [],
"VPCZoneIdentifier": "subnet-5ea0c127,subnet-6194ea3b,subnet-c934b782",
"EnabledMetrics": [],
"Tags": [
          {
 "ResourceId": "group-1",
 "ResourceType": "auto-scaling-group",
 "Key": "environment",
 "Value": "production",
 "PropagateAtLaunch": true
          }
       ],
"TerminationPolicies": [
    "Default"
],
"NewInstancesProtectedFromScaleIn": false,
"ServiceLinkedRoleARN": "arn",
      "TrafficSources": []
   },
```

```
    ... additional groups ...

]
```

Como alternativa, para eliminar todo excepto los nombres de los grupos de escalado automático con los nombres de sus respectivas configuraciones de lanzamiento y etiquetas en el resultado, ejecute el siguiente comando:

```
aws autoscaling describe-auto-scaling-groups --region REGION \
  --query 'AutoScalingGroups[?LaunchConfigurationName!=`null`].{AutoScalingGroupName:
 AutoScalingGroupName, LaunchConfigurationName: LaunchConfigurationName, Tags: Tags}'
```

A continuación se muestra un resultado de ejemplo.

```
[
    {
        "AutoScalingGroupName": "group-1",
        "LaunchConfigurationName": "my-launch-config",
        "Tags": [
            {
                "ResourceId": "group-1",
                "ResourceType": "auto-scaling-group",
                "Key": "environment",
                "Value": "production",
                "PropagateAtLaunch": true
            }
        ]
    },

    ... additional groups ...

]
```

Para obtener más información sobre el filtrado, consulte Filtrar los AWS CLI resultados en la Guía del AWS Command Line Interface usuario.

Paso 2: copiar una configuración de lanzamiento en una plantilla de lanzamiento

Puede copiar una configuración de lanzamiento en una plantilla de lanzamiento mediante el siguiente procedimiento. A continuación, puede agregarlo a su grupo de escalado automático.

Si se copian varias configuraciones de lanzamiento, se obtienen plantillas de lanzamiento con nombres idénticos. Para cambiar el nombre dado a una plantilla de lanzamiento durante el proceso de copia, debe copiar las configuraciones de lanzamiento una por una.

> **Note**
>
> La característica de copia solo está disponible en la consola.

Para copiar una configuración de lanzamiento en una plantilla de lanzamiento (consola)

1. Abra la consola de Amazon EC2 en https://console.aws.amazon.com/ec2/.
2. En el panel de navegación izquierdo, en Escalado automático, elija Grupos de escalado automático.
3. Elija Configuraciones de lanzamiento cerca de la parte superior de la página. Cuando se le pida confirmación, elija Ver configuraciones de lanzamiento para confirmar que desea ver la página Configuraciones de lanzamiento.
4. Seleccione la configuración de lanzamiento que desea copiar y elija Copy to launch template, Copy selected (Copiar en plantilla de lanzamiento, Copiar seleccionada). Se creará una nueva plantilla de lanzamiento con el mismo nombre y opciones que la configuración de lanzamiento que ha seleccionado.
5. En New launch template name (Nombre de la nueva plantilla de lanzamiento), puede utilizar el nombre de la configuración de lanzamiento (el valor predeterminado) o escribir un nuevo nombre. Los nombres de las plantillas de lanzamiento deben ser únicos.
6. (Opcional) Seleccione Crear un grupo de escalado automático utilizando la nueva plantilla.

   Puede omitir este paso para terminar de copiar la configuración de inicio. No es necesario crear un nuevo grupo de escalado automático.
7. Elija Copiar.

Para copiar todas las configuraciones de lanzamiento en plantillas de lanzamiento (consola)

1. Abra la consola de Amazon EC2 en https://console.aws.amazon.com/ec2/.
2. En el panel de navegación, en Auto Scaling, elija Launch Configurations (Configuraciones de lanzamiento).

3. Elija Copy to launch template, Copy all (Copiar a plantilla de lanzamiento, Copiar todo). Se copia cada configuración de lanzamiento en la Región actual en una nueva plantilla de lanzamiento con el mismo nombre y opciones.
4. Elija Copiar.

Paso 3: actualizar un grupo de escalado automático para utilizar una plantilla de lanzamiento

Después de crear una plantilla de lanzamiento, estará listo para agregarla al grupo de escalado automático.

Actualización de un grupo de escalado automático para utilizar una plantilla de lanzamiento (consola)

1. Abra la consola Amazon EC2 en https://console.aws.amazon.com/ec2/y seleccione Auto Scaling Groups en el panel de navegación.
2. Seleccione la casilla situada junto al grupo de escalado automático.

   Se abre un panel dividido en la parte inferior de la página, que muestra información sobre el grupo seleccionado.
3. En la pestaña Details (Detalles), elija Launch configurations (Configuraciones de lanzamiento), Edit (Editar).
4. Elija Switch to launch template (Cambiar a una plantilla de lanzamiento).
5. En Launch template (Plantilla de lanzamiento), seleccione su plantilla de lanzamiento.
6. En Version (Versión), seleccione la versión de la plantilla de lanzamiento que desee. Después de crear versiones de una plantilla de lanzamiento, puede decidir si el grupo de Auto Scaling utilizará la versión predeterminada o la última versión de la plantilla de lanzamiento cuando se realice el escalado horizontal.
7. Elija Actualizar.

Para actualizar un grupo de Auto Scaling para que utilice una plantilla de lanzamiento (AWS CLI)

El siguiente comando update-auto-scaling-group actualiza el grupo de escalado automático especificado para utilizar la versión inicial de la plantilla de lanzamiento especificada.

```
aws autoscaling update-auto-scaling-group --auto-scaling-group-name my-asg \
  --launch-template LaunchTemplateName=my-template-for-auto-scaling,Version='1'
```

Para obtener más ejemplos de uso de comandos CLI para actualizar un grupo de escalado automático para utilizar una plantilla de lanzamiento, consulte Actualización de un grupo de Auto Scaling para utilizar una plantilla de lanzamiento.

Paso 4: reemplazar sus instancias

Después de reemplazar la configuración de lanzamiento por una plantilla de lanzamiento, las nuevas instancias usarán la nueva plantilla de lanzamiento. Las instancias existentes no se ven afectadas.

Para actualizar las instancias existentes, puede iniciar una actualización de instancias para reemplazar las instancias del grupo de escalado automático en lugar de reemplazar manualmente algunas instancias a la vez. Para obtener más información, consulte Uso de una actualización de instancias para actualizar instancias en un grupo de escalado automático. Si el grupo es grande, una actualización de instancias puede ser particularmente útil.

Como alternativa, puede permitir el escalado automático para reemplazar gradualmente las instancias existentes por instancias nuevas basadas en las políticas de terminación del grupo, o puede terminarlas usted. La terminación manual obliga al grupo de escalado automático a lanzar nuevas instancias para mantener la capacidad deseada del grupo. Para obtener más información, consulte Finalizar una instancia en la Guía del usuario de Amazon EC2.

Información adicional

Para obtener más información, consulte Amazon EC2 Auto Scaling dejará de añadir soporte para las nuevas funciones de EC2 a las configuraciones de lanzamiento en AWS el blog de informática.

Para ver un tema que explica cómo migrar AWS CloudFormation pilas de configuraciones de lanzamiento a plantillas de lanzamiento, consulte. Migración AWS CloudFormation pilas para lanzar plantillas

Migración AWS CloudFormation pilas para lanzar plantillas

Puede migrar sus plantillas de CloudFormation pila existentes de configuraciones de lanzamiento a plantillas de lanzamiento. Para ello, agregue una plantilla de lanzamiento directamente a una plantilla de pila existente y, a continuación, asocie la plantilla de lanzamiento al grupo de escalado automático de la plantilla de pila. A continuación, utilice la plantilla modificada para actualizar su pila.

Al migrar a plantillas de lanzamiento, este tema le permite ahorrar tiempo, ya que proporciona instrucciones para reescribir las configuraciones de lanzamiento de las plantillas de CloudFormation pila como plantillas de lanzamiento. Para obtener más información sobre la migración de

configuraciones de lanzamiento a plantillas de lanzamiento, consulte Migración de sus grupos de escalado automático a plantillas de lanzamiento.

Temas

- Búsqueda de grupos de escalado automático que utilizan una configuración de lanzamiento
- Actualización de una pila para utilizar una plantilla de lanzamiento
- Comprensión del comportamiento de actualización de los recursos de la pila
- Seguimiento de la migración
- Referencia de mapeo de configuración de lanzamiento

Búsqueda de grupos de escalado automático que utilizan una configuración de lanzamiento

Búsqueda de grupos de escalado automático que utilizan una configuración de lanzamiento

- Utilice el comando describe-auto-scaling-groups para enumerar los nombres de los grupos de escalado automático que utilizan configuraciones de lanzamiento en la región especificada. Incluye la `--filters` opción de restringir los resultados a los grupos asociados a una CloudFormation pila (filtrándolos por la clave de la `aws:cloudformation:stack-name` etiqueta).

  ```
  aws autoscaling describe-auto-scaling-groups --region REGION \
    --filters Name=tag-key,Values=aws:cloudformation:stack-name \
    --query 'AutoScalingGroups[?LaunchConfigurationName!
  =`null`].AutoScalingGroupName'
  ```

  A continuación se muestra un resultado de ejemplo.

  ```
  [
      "{stack-name}-group-1",
      "{stack-name}-group-2",
      "{stack-name}-group-3"
  ]
  ```

  Puede encontrar AWS CLI comandos útiles adicionales para buscar grupos de Auto Scaling para migrarlos y filtrar la salidaMigración de sus grupos de escalado automático a plantillas de lanzamiento.

> **Important**
>
> Si los recursos de tu pila tienen `AWSEB` su nombre, significa que se crearon mediante AWS Elastic Beanstalk. En este caso, debe actualizar el entorno de Beanstalk para indicar a Elastic Beanstalk que elimine la configuración de lanzamiento y la sustituya por una plantilla de lanzamiento.

Actualización de una pila para utilizar una plantilla de lanzamiento

Siga los pasos de esta sección para hacer lo siguiente:

- Reescriba la configuración de lanzamiento como una plantilla de lanzamiento utilizando las propiedades de la plantilla de lanzamiento equivalentes.
- Asocie la nueva plantilla de lanzamiento con el grupo de escalado automático.
- Implemente estas actualizaciones.

Modificación de la plantilla de pila y actualizar la pila

1. Siga los mismos procedimientos generales para modificar la plantilla de pila descritos en Modificación de una plantilla de pila de la Guía del usuario de AWS CloudFormation .
2. Reescriba la configuración de lanzamiento como una plantilla de lanzamiento. Vea el siguiente ejemplo:

   Ejemplo: una configuración de lanzamiento sencilla

```
---
Resources:
  myLaunchConfig:
    Type: AWS::AutoScaling::LaunchConfiguration
    Properties:
      ImageId: ami-02354e95b3example
      InstanceType: t3.micro
      SecurityGroups:
        - !Ref EC2SecurityGroup
      KeyName: MyKeyPair
      BlockDeviceMappings:
        - DeviceName: /dev/xvda
          Ebs:
```

```
            VolumeSize: 150
            DeleteOnTermination: true
      UserData:
        Fn::Base64: !Sub |
          #!/bin/bash -xe
          yum install -y aws-cfn-bootstrap
          /opt/aws/bin/cfn-signal -e $? --stack ${AWS::StackName} --resource myASG
 --region ${AWS::Region}
```

Ejemplo: la plantilla de lanzamiento equivalente

```
---
Resources:
  myLaunchTemplate:
    Type: AWS::EC2::LaunchTemplate
    Properties:
      LaunchTemplateName: !Sub ${AWS::StackName}-launch-template
      LaunchTemplateData:
        ImageId: ami-02354e95b3example
        InstanceType: t3.micro
        SecurityGroupIds:
          - Ref! EC2SecurityGroup
        KeyName: MyKeyPair
        BlockDeviceMappings:
          - DeviceName: /dev/xvda
            Ebs:
              VolumeSize: 150
              DeleteOnTermination: true
        UserData:
          Fn::Base64: !Sub |
            #!/bin/bash -x
            yum install -y aws-cfn-bootstrap
            /opt/aws/bin/cfn-signal -e $? --stack ${AWS::StackName} --resource
 myASG --region ${AWS::Region}
```

Para obtener información de referencia sobre todas las propiedades compatibles con Amazon EC2, consulte AWS: :EC2 :: LaunchTemplate AWS: :EC2 en la Guía del usuario. LaunchTemplate AWS CloudFormation

Observe cómo la plantilla de lanzamiento incluye la propiedad `LaunchTemplateName` con un valor de `!Sub ${AWS::StackName}-launch-template`. Esto es obligatorio si desea que el nombre de la plantilla de lanzamiento incluya el nombre de la pila.

3. Si la propiedad **`IamInstanceProfile`** está presente en su configuración de lanzamiento, tendrá que convertirla en una estructura y especificar el nombre o el ARN del perfil de instancia. Para ver un ejemplo, consulte AWS: :EC2:: LaunchTemplate AWS: . LaunchTemplate

4. Si las propiedades **`AssociatePublicIpAddress`**, **`InstanceMonitoring`** o **`PlacementTenancy`** están presentes en la configuración de lanzamiento, debe convertirlas en una estructura. Para ver ejemplos, consulte AWS: :EC2:: LaunchTemplate AWS: . LaunchTemplate

   Se produce una excepción cuando el valor de la propiedad `MapPublicIpOnLaunch` en las subredes que utilizó para su grupo de escalado automático coincide con el valor de la propiedad `AssociatePublicIpAddress` en su configuración de lanzamiento. En este caso, puede ignorar la propiedad `AssociatePublicIpAddress`. La propiedad `AssociatePublicIpAddress` solo se usa para anular la propiedad `MapPublicIpOnLaunch` y cambiar si las instancias reciben una dirección IPv4 pública en el momento del lanzamiento.

5. Puede copiar los grupos de seguridad de la propiedad **`SecurityGroups`** en uno de los dos lugares de la plantilla de lanzamiento. Normalmente, los grupos de seguridad se copian en la propiedad `SecurityGroupIds`. Sin embargo, si crea una estructura de `NetworkInterfaces` en la plantilla de lanzamiento para especificar la propiedad `AssociatePublicIpAddress`, deberá copiar los grupos de seguridad en la propiedad `Groups` de la interfaz de red.

6. Si hay alguna estructura de `BlockDeviceMapping` en la configuración de lanzamiento con el valor **`NoDevice`** establecido en `true`, debe especificar una cadena vacía para `NoDevice` en su plantilla de lanzamiento para que Amazon EC2 omita el dispositivo.

7. Si la propiedad **`SpotPrice`** está presente en su configuración de lanzamiento, le recomendamos que la omita de la plantilla de lanzamiento. Las instancias de spot se lanzarán al precio de spot actual. Este precio nunca superará el precio. On-Demand

   Para solicitar instancias de spot, tiene dos opciones que se excluyen mutuamente:

   - La primera consiste en utilizar la estructura de `InstanceMarketOptions` en la plantilla de lanzamiento (no se recomienda). Para obtener más información, consulte AWS: :EC2:: LaunchTemplate InstanceMarketOptions AWS: en la Guía del usuario.AWS CloudFormation
   - La otra es agregar una estructura de `MixedInstancesPolicy` al grupo de escalado automático. De este modo, dispondrá de más opciones para realizar la solicitud. Una solicitud de instancia de spot en su plantilla de lanzamiento no admite más de una selección de tipo de instancia por grupo de escalado automático. Sin embargo, una política de instancias mixtas admite la selección de más de un tipo de instancia por grupo de escalado automático. Las

solicitudes de instancias de spot se benefician de tener más de un tipo de instancia entre los que elegir. Para obtener más información, consulte AWS::AutoScaling::: AutoScalingGroup MixedInstancesPolicy en la Guía del usuario.AWS CloudFormation

8. Elimine la **`LaunchConfigurationName`** propiedad del AutoScalingGroup recurso AWS::AutoScaling: AutoScalingGroup AWS::AutoScaling . Agregue la plantilla de lanzamiento en su lugar.

   En los siguientes ejemplos, la función intrínseca Ref obtiene el ID del LaunchTemplate recurso AWS: :EC2:: LaunchTemplate AWS: con el ID lógico. `myLaunchTemplate` La GetAttfunción obtiene el número de versión más reciente (por ejemplo,`1`) de la plantilla de lanzamiento de la propiedad. `Version`

   Ejemplo: sin una política de instancias mixtas

   ```
   ---
   Resources:
     myASG:
       Type: AWS::AutoScaling::AutoScalingGroup
       Properties:
         LaunchTemplate:
           LaunchTemplateId: !Ref myLaunchTemplate
           Version: !GetAtt myLaunchTemplate.LatestVersionNumber
         ...
   ```

   Ejemplo: con una política de instancias mixtas

   ```
   ---
   Resources:
     myASG:
       Type: AWS::AutoScaling::AutoScalingGroup
       Properties:
         MixedInstancesPolicy:
           LaunchTemplate:
             LaunchTemplateSpecification:
               LaunchTemplateId: !Ref myLaunchTemplate
               Version: !GetAtt myLaunchTemplate.LatestVersionNumber
         ...
   ```

Para obtener información de referencia sobre todas las propiedades que admite Amazon EC2 Auto Scaling, AWS::AutoScalingconsulte::: AutoScalingGroup AWS::AutoScaling en AWS CloudFormation la Guía del usuario.

9. Cuando esté listo para implementar estas actualizaciones, siga los CloudFormation procedimientos para actualizar la pila con la plantilla de pila modificada. Para obtener más información, consulte Modificación de una plantilla de pila en la AWS CloudFormation Guía del usuario de .

Comprensión del comportamiento de actualización de los recursos de la pila

CloudFormation actualiza los recursos de la pila comparando los cambios entre la plantilla actualizada que proporciona y las configuraciones de recursos que describió en la versión anterior de la plantilla de pila. Las configuraciones de recursos que no han cambiado no se ven afectadas durante el proceso de actualización.

CloudFormation admite el UpdatePolicyatributo para los grupos de Auto Scaling. Para reemplazar las instancias existentes después de completar este procedimiento, añada un `UpdatePolicy` atributo a la pila. Defina la política de actualización en`AutoScalingInstanceRefresh`. Para obtener más información, consulte la AutoScalingInstanceRefresh política en la Guía AWS CloudFormation del usuario.

Si no especificó un `UpdatePolicy` atributo para su grupo de Auto Scaling, se comprueba que la plantilla de lanzamiento sea correcta, pero CloudFormation no implementa ningún cambio en las instancias del grupo Auto Scaling. Todas las instancias nuevas usarán su plantilla de lanzamiento, pero las instancias existentes continuarán ejecutándose con la configuración de lanzamiento con la que se lanzaron la primera vez (a pesar de que no exista configuración de lanzamiento). La excepción se produce cuando cambia las opciones de compra, por ejemplo, agregando una política de instancias mixtas. En este caso, su grupo de escalado automático reemplaza gradualmente las instancias existentes por instancias nuevas para que coincidan con las nuevas opciones de compra.

Si tiene que deshacer un cambio para pasar de las configuraciones de lanzamiento a las plantillas de lanzamiento, asegúrese de probar la operación de reversión.

Seguimiento de la migración

Seguimiento de la migración

1. En la consola de CloudFormation, seleccione la pila que ha actualizado y, a continuación, elija la pestaña Events (Eventos) para ver los eventos de pila.
2. Para actualizar la lista de eventos con los eventos más recientes, pulse el botón de actualización de la CloudFormation consola.
3. Mientras se actualiza su pila, observará varios eventos por cada actualización de recursos. Si ve una excepción en la columna Motivo del estado que indica que hay un problema al intentar crear la plantilla de lanzamiento, consulte Solución de problemas de Amazon EC2 Auto Scaling: plantillas de lanzamiento para ver las posibles causas.
4. (Opcional) Según el uso que haga del atributo `UpdatePolicy`, puede supervisar el progreso del grupo de escalado automático desde la página de grupos de escalado automático de la consola de Amazon EC2. Seleccione el grupo de Auto Scaling. En la pestaña Activity (Actividad), en Activity history (Historial de actividad), la columna Status (Estado) muestra si su grupo de escalado automático ha lanzado las instancias o las ha terminado correctamente, o bien si la actividad de escalado sigue en curso.
5. Cuando se complete la actualización de la pila, CloudFormation emite un evento de `UPDATE_COMPLETE` pila. Para obtener más información, consulte Monitorización del progreso de una actualización de pila en la Guía de usuario de AWS CloudFormation .
6. Una vez finalizada la actualización de la pila, abra la Página de plantillas de lanzamiento y la Página de configuraciones de lanzamiento de la consola de Amazon EC2. Observará que se ha creado una nueva plantilla de lanzamiento y que se ha eliminado la configuración de lanzamiento.

Referencia de mapeo de configuración de lanzamiento

Como referencia, en la siguiente tabla se enumeran todas las propiedades de nivel superior del recurso AWS::AutoScaling:: LaunchConfiguration AWS::AutoScaling con su propiedad correspondiente en el LaunchConfiguration recurso AWS: :EC2:: LaunchTemplate AWS: . LaunchTemplate

Propiedad fuente de configuración de lanzamiento	Propiedad de objetivo de la plantilla de lanzamiento
`AssociatePublicIpAddress`	`NetworkInterfaces.AssociatePublicIpAddress`
`BlockDeviceMappings`	`BlockDeviceMappings`
`ClassicLinkVPCId`	No disponible[1]
`ClassicLinkVPCSecurityGroups`	No disponible[1]
`EbsOptimized`	`EbsOptimized`
`IamInstanceProfile`	`IamInstanceProfile.Arn` o `IamInstanceProfile.Name`, pero no ambos
`ImageId`	`ImageId`
`InstanceId`	`InstanceId`
`InstanceMonitoring`	`Monitoring.Enabled`
`InstanceType`	`InstanceType`
`KernelId`	`KernelId`
`KeyName`	`KeyName`
`LaunchConfigurationName`	`LaunchTemplateName`
`MetadataOptions`	`MetadataOptions`
`PlacementTenancy`	`Placement.Tenancy`
`RamDiskId`	`RamDiskId`
`SecurityGroups`	`SecurityGroupIds` o `NetworkInterfaces.Groups`, pero no ambos

Propiedad fuente de configuración de lanzamiento	Propiedad de objetivo de la plantilla de lanzamiento
`SpotPrice`	`InstanceMarketOptions.SpotOptions.MaxPrice`
`UserData`	`UserData`

[1] Las `ClassicLinkVPCSecurityGroups` propiedades `ClassicLinkVPCId` y no están disponibles para su uso en una plantilla de lanzamiento porque ya EC2-Classic no están disponibles.

Ejemplos de creación y administración de plantillas de lanzamiento con AWS CLI

Puede crear y administrar plantillas de lanzamiento a través de los Consola de administración de AWS, AWS Command Line Interface (AWS CLI) o los SDK. En esta sección se muestran ejemplos de creación y administración de plantillas de lanzamiento para Amazon EC2 Auto Scaling desde AWS CLI.

Contenido

- Ejemplo de uso
- Creación de una plantilla de lanzamiento básica
- Especificar etiquetas que etiquetan instancias en el lanzamiento
- Especificar un rol de IAM para transferir a instancias
- Asignación de direcciones IP públicas
- Especificar un script de datos de usuario que configura instancias en el lanzamiento
- Especificar una asignación de dispositivos de bloques
- Especificar hosts dedicados para traer licencias de software de proveedores externos
- Especificar una interfaz de red existente
- Creación de múltiples interfaces de red
- Administración de las plantillas de lanzamiento
- Actualización de un grupo de Auto Scaling para utilizar una plantilla de lanzamiento

Ejemplo de uso

```
{
    "LaunchTemplateName": "my-template-for-auto-scaling",
    "VersionDescription": "test description",
    "LaunchTemplateData": {
        "ImageId": "ami-04d5cc9b88example",
        "InstanceType": "t2.micro",
        "SecurityGroupIds": [
            "sg-903004f88example"
        ],
        "KeyName": "MyKeyPair",
        "Monitoring": {
            "Enabled": true
        },
        "Placement": {
            "Tenancy": "dedicated"
        },
        "CreditSpecification": {
            "CpuCredits": "unlimited"
        },
        "MetadataOptions": {
            "HttpTokens": "required",
            "HttpPutResponseHopLimit": 1,
            "HttpEndpoint": "enabled"
        }
    }
}
```

Creación de una plantilla de lanzamiento básica

Para crear una plantilla de lanzamiento básica, utilice el comando create-launch-template de la siguiente manera, con estas modificaciones:

- Reemplace `ami-04d5cc9b88example` con el ID de la AMI desde la que se lanzan las instancias.
- Reemplace `t2.micro` con un tipo de instancia que sea compatible con la AMI que especificó.

En este ejemplo, se crea una plantilla de lanzamiento con el nombre *`my-template-for-auto-scaling`*. Si las instancias creadas por esta plantilla de inicio se inician en una VPC predeterminada,

recibirán 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.

```
aws ec2 create-launch-template --launch-template-name my-template-for-auto-scaling --
version-description version1 \
  --launch-template-data
 '{"ImageId":"ami-04d5cc9b88example","InstanceType":"t2.micro"}'
```

Para obtener más información sobre JSON-formatted los parámetros de cotización, consulte Uso de comillas con cadenas AWS CLI en la Guía del AWS Command Line Interface usuario.

Como alternativa, puede especificar los JSON-formatted parámetros en un archivo de configuración.

En el ejemplo siguiente se crea una plantilla de lanzamiento básica, que hace referencia a un archivo de configuración para los valores de parámetro de plantilla de lanzamiento.

```
aws ec2 create-launch-template --launch-template-name my-template-for-auto-scaling --
version-description version1 \
  --launch-template-data file://config.json
```

Contenido de `config.json`:

```
{
    "ImageId":"ami-04d5cc9b88example",
    "InstanceType":"t2.micro"
}
```

Especificar etiquetas que etiquetan instancias en el lanzamiento

En el siguiente ejemplo se agrega una etiqueta (por ejemplo, `purpose=webserver`) a instancias en el lanzamiento.

```
aws ec2 create-launch-template --launch-template-name my-template-for-auto-scaling --
version-description version1 \
  --launch-template-data '{"TagSpecifications":[{"ResourceType":"instance","Tags":
[{"Key":"purpose","Value":"webserver"}]}],"ImageId":"ami-04d5cc9b88example","InstanceType":"t2.
```

> **Note**
>
> Si especifica etiquetas de instancia en la plantilla de lanzamiento y elige propagar las etiquetas del grupo de Auto Scaling a sus instancias, todas las etiquetas se fusionarán. Si se especifica la misma clave de etiqueta para una etiqueta en su plantilla de lanzamiento y una etiqueta en su grupo de Auto Scaling, entonces el valor de la etiqueta del grupo tiene prioridad.

Especificar un rol de IAM para transferir a instancias

En el ejemplo siguiente se especifica el nombre del perfil de instancia asociado con el rol de IAM que se va a transferir a las instancias en el lanzamiento. Para obtener más información, consulte Rol de IAM para aplicaciones que se ejecuten en instancias de Amazon EC2.

```
aws ec2 create-launch-template --launch-template-name my-template-for-auto-scaling --
version-description version1 \
--launch-template-data '{"IamInstanceProfile":{"Name":"my-instance-
profile"},"ImageId":"ami-04d5cc9b88example","InstanceType":"t2.micro"}'
```

Asignación de direcciones IP públicas

El ejemplo siguiente create-launch-template configura una plantilla de lanzamiento para asignar direcciones públicas a instancias lanzadas en una VPC no predeterminada.

> **Note**
>
> Cuando especifique una interfaz de red, especifique un valor para `Groups` que se corresponda con los grupos de seguridad de la VPC en la que el grupo de Auto Scaling lanzará las instancias. Especifique las subredes de la VPC como propiedades del grupo de Auto Scaling.

```
aws ec2 create-launch-template --launch-template-name my-template-for-auto-scaling --
version-description version1 \
  --launch-template-data '{"NetworkInterfaces":
[{"DeviceIndex":0,"AssociatePublicIpAddress":true,"Groups":
["sg-903004f88example"],"DeleteOnTermination":true}],"ImageId":"ami-04d5cc9b88example","Instanc
```

Especificar un script de datos de usuario que configura instancias en el lanzamiento

En el ejemplo siguiente se especifica una secuencia de comandos de datos de usuario como una cadena codificada en base64 que configura instancias en el momento del lanzamiento. El comando create-launch-template requiere datos de usuario codificados en base64.

```
aws ec2 create-launch-template --launch-template-name my-template-for-auto-scaling --
version-description version1 \
--launch-template-data
 '{"UserData":"IyEvYmluL2Jhc...","ImageId":"ami-04d5cc9b88example","InstanceType":"t2.micro"}'
```

Especificar una asignación de dispositivos de bloques

El ejemplo siguiente create-launch-template crea una plantilla de lanzamiento con una asignación de dispositivos de bloques: un volumen EBS cifrado de 22 gigabytes asignado a `/dev/xvdcz`. El volumen `/dev/xvdcz` utiliza el tipo de volumen SSD de uso general (gp2) y se elimina al finalizar la instancia a la que está adjunta.

```
aws ec2 create-launch-template --launch-template-name my-template-for-auto-scaling --
version-description version1 \
  --launch-template-data '{"BlockDeviceMappings":[{"DeviceName":"/dev/xvdcz","Ebs":
{"VolumeSize":22,"VolumeType":"gp2","DeleteOnTermination":true}}],"ImageId":"ami-04d5cc9b88exam
```

Especificar hosts dedicados para traer licencias de software de proveedores externos

Si especifica la tenencia de host, puede especificar un grupo de recursos de host y una configuración de licencias de License Manager para traer licencias de software elegibles de proveedores externos. A continuación, puede utilizar las licencias en instancias EC2 mediante el siguiente comando create-launch-template.

```
aws ec2 create-launch-template --launch-template-name my-template-for-auto-scaling --
version-description version1 \
  --launch-template-data '{"Placement":
{"Tenancy":"host","HostResourceGroupArn":"arn"},"LicenseSpecifications":
[{"LicenseConfigurationArn":"arn"}],"ImageId":"ami-04d5cc9b88example","InstanceType":"t2.micro"
```

Especificar una interfaz de red existente

El siguiente ejemplo create-launch-template configura la interfaz de red principal para que utilice una interfaz de red existente.

```
aws ec2 create-launch-template --launch-template-name my-template-for-auto-scaling --
version-description version1 \
  --launch-template-data '{"NetworkInterfaces":
[{"DeviceIndex":0,"NetworkInterfaceId":"eni-
b9a5ac93","DeleteOnTermination":false}],"ImageId":"ami-04d5cc9b88example","InstanceType":"t2.mi
```

Creación de múltiples interfaces de red

El siguiente ejemplo create-launch-template agrega una interfaz de red secundaria. La interfaz de red principal tiene un índice de dispositivo de 0 y la interfaz de red secundaria tiene un índice de dispositivo de 1.

```
aws ec2 create-launch-template --launch-template-name my-template-for-auto-scaling --
version-description version1 \
  --launch-template-data '{"NetworkInterfaces":[{"DeviceIndex":0,"Groups":
["sg-903004f88example"],"DeleteOnTermination":true},{"DeviceIndex":1,"Groups":
["sg-903004f88example"],"DeleteOnTermination":true}],"ImageId":"ami-04d5cc9b88example","Instanc
```

Si utiliza un tipo de instancia que admita varias tarjetas de red y Elastic Fabric Adapters (EFA), puede agregar una interfaz secundaria a una tarjeta de red secundaria y habilitar EFA mediante el siguiente comando create-launch-template. Para obtener más información, consulte Agregar un EFA a una plantilla de lanzamiento en la Guía del usuario de Amazon EC2.

```
aws ec2 create-launch-template --launch-template-name my-template-for-auto-scaling --
version-description version1 \
  --launch-template-data '{"NetworkInterfaces":
[{"NetworkCardIndex":0,"DeviceIndex":0,"Groups":
["sg-7c2270198example"],"InterfaceType":"efa","DeleteOnTermination":true},
{"NetworkCardIndex":1,"DeviceIndex":1,"Groups":
["sg-7c2270198example"],"InterfaceType":"efa","DeleteOnTermination":true}],"ImageId":"ami-09d95
```

> **Warning**
>
> El tipo de instancia p4d.24xlarge incurre en costos más altos que los otros ejemplos de esta sección. Para obtener más información acerca de los precios para instancias P4d, consulte Precios de instancias P4d de Amazon EC2.

> **Note**
>
> Si se asocian varias interfaces de red de la misma subred a una instancia, se puede presentar un enrutamiento asimétrico, especialmente en instancias que utilizan una variante de Amazon Linux. Si necesita este tipo de configuración, debe configurar la interfaz de red secundaria dentro del sistema operativo. Para ver un ejemplo, consulte ¿Cómo puedo hacer que mi interfaz de red secundaria funcione en mi instancia EC2 de Ubuntu? en el Centro de AWS conocimiento.

Administración de las plantillas de lanzamiento

AWS CLI Incluye varios otros comandos que le ayudan a administrar sus plantillas de lanzamiento.

Contenido

- Enumeración y descripción de las plantillas de lanzamiento
- Crear una versión de plantilla de inicialización
- Eliminar una versión de plantilla de inicialización
- Eliminación de una plantilla de inicialización

Enumeración y descripción de las plantillas de lanzamiento

Puedes usar dos AWS CLI comandos para obtener información sobre tus plantillas de lanzamiento: describe-launch-templates y describe-launch-template-versions.

El comando describe-launch-templates le permite obtener una lista de cualquiera de las plantillas de lanzamiento que ha creado. Puede utilizar una opción para filtrar los resultados en un nombre de plantilla de lanzamiento, crear tiempo, clave de etiqueta o combinación clave-valor de etiqueta. Este comando devuelve información resumida sobre cualquiera de sus plantillas de lanzamiento, incluido el identificador de plantilla de lanzamiento, la versión más reciente y la versión predeterminada.

En el ejemplo siguiente se proporciona un resumen de la plantilla de lanzamiento especificada.

```
aws ec2 describe-launch-templates --launch-template-names my-template-for-auto-scaling
```

A continuación, se muestra un ejemplo de respuesta.

```
{
    "LaunchTemplates": [
        {
            "LaunchTemplateId": "lt-068f72b729example",
            "LaunchTemplateName": "my-template-for-auto-scaling",
            "CreateTime": "2020-02-28T19:52:27.000Z",
            "CreatedBy": "arn:aws:iam::123456789012:user/Bob",
            "DefaultVersionNumber": 1,
            "LatestVersionNumber": 1
        }
    ]
}
```

Si no utiliza `--launch-template-names` para limitar la salida a una plantilla de lanzamiento, se devuelve información sobre todas las plantillas de lanzamiento.

El siguiente comando describe-launch-template-versions proporciona información que describe las versiones de la plantilla de lanzamiento especificada.

```
aws ec2 describe-launch-template-versions --launch-template-id lt-068f72b729example
```

A continuación, se muestra un ejemplo de respuesta.

```
{
    "LaunchTemplateVersions": [
        {
            "VersionDescription": "version1",
            "LaunchTemplateId": "lt-068f72b729example",
            "LaunchTemplateName": "my-template-for-auto-scaling",
            "VersionNumber": 1,
            "CreatedBy": "arn:aws:iam::123456789012:user/Bob",
            "LaunchTemplateData": {
                "TagSpecifications": [
                    {
                        "ResourceType": "instance",
                        "Tags": [
```

```
                        {
                            "Key": "purpose",
                            "Value": "webserver"
                        }
                    ]
                }
            ],
            "ImageId": "ami-04d5cc9b88example",
            "InstanceType": "t2.micro",
            "NetworkInterfaces": [
                {
                    "DeviceIndex": 0,
                    "DeleteOnTermination": true,
                    "Groups": [
                        "sg-903004f88example"
                    ],
                    "AssociatePublicIpAddress": true
                }
            ]
        },
        "DefaultVersion": true,
        "CreateTime": "2020-02-28T19:52:27.000Z"
    }
]
}
```

Crear una versión de plantilla de inicialización

El siguiente comando create-launch-template crea una nueva versión de plantilla de lanzamiento basada en la versión 1 de la plantilla de lanzamiento y especifica otro ID de AMI.

```
aws ec2 create-launch-template-version --launch-template-id lt-068f72b729example --
version-description version2 \
  --source-version 1 --launch-template-data "ImageId=ami-c998b6b2example"
```

Para establecer la versión predeterminada de la plantilla de lanzamiento, utilice el comando modificy-launch-template.

Eliminar una versión de plantilla de inicialización

El siguiente comando delete-launch-template-versions elimina la versión especificada de una plantilla de lanzamiento.

```
aws ec2 delete-launch-template-versions --launch-template-id lt-068f72b729example --
versions 1
```

Eliminación de una plantilla de inicialización

Si ya no necesita una plantilla de lanzamiento, puede eliminarla mediante el siguiente comando delete launch-template. Al eliminar una plantilla de inicialización, también se eliminan todas sus versiones.

```
aws ec2 delete-launch-template --launch-template-id lt-068f72b729example
```

Actualización de un grupo de Auto Scaling para utilizar una plantilla de lanzamiento

Puede utilizar el coamdo update-auto-scaling-group para agregar una plantilla de lanzamiento a un grupo de Auto Scaling existente.

Actualización de un grupo de Auto Scaling para utilizar la versión más reciente de una plantilla de lanzamiento

El siguiente comando update-auto-scaling-group actualiza el grupo de Auto Scaling especificado para utilizar la última versión de la plantilla de lanzamiento especificada.

```
aws autoscaling update-auto-scaling-group --auto-scaling-group-name my-asg \
  --launch-template LaunchTemplateId=lt-068f72b729example,Version='$Latest'
```

Actualización de un grupo de Auto Scaling para utilizar una versión específica de una plantilla de lanzamiento

El siguiente comando update-auto-scaling-group actualiza el grupo de Auto Scaling especificado para utilizar una versión específica de la plantilla de lanzamiento especificada.

```
aws autoscaling update-auto-scaling-group --auto-scaling-group-name my-asg \
  --launch-template LaunchTemplateName=my-template-for-auto-scaling,Version='2'
```

Uso AWS Systems Manager parámetros en lugar de ID de AMI en las plantillas de lanzamiento

En esta sección, se muestra cómo crear una plantilla de lanzamiento que especifique un parámetro de AWS Systems Manager que haga referencia a un ID de imagen de máquina de Amazon (AMI). Puede usar un parámetro almacenado en su propia AMI Cuenta de AWS, un parámetro compartido con otro Cuenta de AWS o un parámetro público para una AMI pública mantenida por AWS.

Con los parámetros de Systems Manager, puede actualizar los grupos de escalado automático para utilizar nuevos ID de AMI sin necesidad de crear nuevas plantillas de lanzamiento o nuevas versiones de estas cada vez que cambie un ID de AMI. Estos ID pueden cambiar con regularidad, como cuando una AMI se actualiza con actualizaciones de software o el sistema operativo más reciente.

Puede crear, actualizar o eliminar sus propios parámetros de Systems Manager mediante el Almacén de parámetros, una capacidad de AWS Systems Manager. Debe crear un parámetro de Systems Manager antes de poder usarlo en una plantilla de lanzamiento. Para comenzar, cree un parámetro con el tipo de datos `aws:ec2:image`, y para su valor, especifique el ID de una AMI. El ID de AMI tiene el formato `ami-<identifier>`, por ejemplo, `ami-123example456`. El ID de AMI correcto depende del tipo de instancia y la Región de AWS en la que quiere lanzar el grupo de escalado automático.

Para obtener más información sobre cómo crear un parámetro válido para un ID de AMI, consulte Creación de parámetros de Systems Manager.

Creación de una plantilla de lanzamiento que especifique un parámetro para la AMI

Para crear una plantilla de lanzamiento que especifique un parámetro para la AMI, utilice uno de los métodos siguientes:

Console

Para crear una plantilla de lanzamiento mediante un AWS Systems Manager parameter

1. Abra la consola de Amazon EC2 en https://console.aws.amazon.com/ec2/.
2. En el panel de navegación, elija Launch Templates (Plantillas de inicialización) y, a continuación, Create launch template (Crear plantilla de inicialización).

3. En Launch template name (Nombre de plantilla de inicialización), introduzca un nombre descriptivo para la plantilla.
4. En Application and OS Images (Imagen de máquina de Amazon) (Imágenes de aplicaciones y sistema operativo [imagen de máquina de Amazon]), elija Browse more AMIs (Buscar más AMI).
5. Pulse el botón de flecha situado a la derecha de la barra de búsqueda y, a continuación, elija Especificar un parámetro value/Systems de administrador personalizado.
6. En el cuadro de diálogo Especificar valor personalizado o parámetro de Systems Manager, haga lo siguiente:

   a. Para el ID de AMI o la cadena de parámetros de Systems Manager, introduzca el nombre del parámetro de Systems Manager mediante uno de los siguientes formatos:

      Para hacer referencia a un parámetro público:

      - **resolve:ssm:*public-parameter***

      Para hacer referencia a un parámetro almacenado en la misma cuenta:

      - **resolve:ssm:*parameter-name***
      - **resolve:ssm:*parameter-name*:*version-number***
      - **resolve:ssm:*parameter-name*:*label***

      Para hacer referencia a un parámetro compartido desde otra Cuenta de AWS:

      - **resolve:ssm:*parameter-ARN***
      - **resolve:ssm:*parameter-ARN*:*version-number***
      - **resolve:ssm:*parameter-ARN*:*label***

   b. Seleccione Save.
7. Configure cualquier otro ajuste de la plantilla de lanzamiento según sea necesario y, a continuación, seleccione Crear plantilla de lanzamiento. Para obtener más información, consulte Creación de una plantilla de lanzamiento para un grupo de Auto Scaling.

AWS CLI

Para crear una plantilla de lanzamiento que especifique un parámetro de Systems Manager, puede utilizar uno de los siguientes comandos de ejemplo. Reemplace cada *`user input placeholder`* por su propia información.

Ejemplo: cree una plantilla de lanzamiento que especifique un AWS-parámetro público de propiedad

Utilice la siguiente sintaxis: `resolve:ssm:`*`public-parameter`*, donde `resolve:ssm` es el prefijo estándar y *`public-parameter`* es la ruta y el nombre del parámetro público.

En este ejemplo, la plantilla de lanzamiento utiliza un parámetro público AWS proporcionado para lanzar instancias con la última AMI de Amazon Linux 2 configurada para su perfil. Región de AWS

```
aws ec2 create-launch-template --launch-template-name my-template-for-auto-scaling
 --version-description version1 \
  --launch-template-data file://config.json
```

Contenido de `config.json`:

```
{
    "ImageId":"resolve:ssm:/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-
x86_64-gp2",
    "InstanceType":"t2.micro"
}
```

A continuación, se muestra un ejemplo de respuesta.

```
{
    "LaunchTemplate": {
        "LaunchTemplateId": "lt-089c023a30example",
        "LaunchTemplateName": "my-template-for-auto-scaling",
        "CreateTime": "2022-12-28T19:52:27.000Z",
        "CreatedBy": "arn:aws:iam::123456789012:user/Bob",
        "DefaultVersionNumber": 1,
        "LatestVersionNumber": 1
    }
}
```

Ejemplo: creación de una plantilla de lanzamiento que especifica un parámetro almacenado en la misma cuenta

Utilice la siguiente sintaxis: `resolve:ssm:`*`parameter-name`*, donde `resolve:ssm` es el prefijo estándar y *`parameter-name`* es el nombre del parámetro de Systems Manager.

En el ejemplo siguiente se crea una plantilla de lanzamiento que obtiene el ID de AMI de un parámetro de Systems Manager existente denominado *`golden-ami`*.

```
aws ec2 create-launch-template --launch-template-name my-template-for-auto-scaling \
  --launch-template-data file://config.json
```

Contenido de `config.json`:

```
{
    "ImageId":"resolve:ssm:golden-ami",
    "InstanceType":"t2.micro"
}
```

La versión predeterminada del parámetro, cuando no se especifica, es la versión más reciente.

El siguiente ejemplo hace referencia a una versión específica del parámetro *`golden-ami`*. El ejemplo usa la versión *`3`* del parámetro *`golden-ami`*, pero puede usar cualquier número de versión válido.

```
{
    "ImageId":"resolve:ssm:golden-ami:3",
    "InstanceType":"t2.micro"
}
```

El siguiente ejemplo similar hace referencia a una etiqueta del parámetro *`prod`* que asigna a una versión específica del parámetro *`golden-ami`*.

```
{
    "ImageId":"resolve:ssm:golden-ami:prod",
    "InstanceType":"t2.micro"
}
```

A continuación, se muestra un ejemplo del resultado.

```
{
    "LaunchTemplate": {
        "LaunchTemplateId": "lt-068f72b724example",
```

```
        "LaunchTemplateName": "my-template-for-auto-scaling",
        "CreateTime": "2022-12-27T17:11:21.000Z",
        "CreatedBy": "arn:aws:iam::123456789012:user/Bob",
        "DefaultVersionNumber": 1,
        "LatestVersionNumber": 1
    }
}
```

Ejemplo: cree una plantilla de lanzamiento que especifique un parámetro compartido con otro Cuenta de AWS

Utilice la siguiente sintaxis: `resolve:ssm:`*`parameter-ARN`*, donde `resolve:ssm` es el prefijo estándar y *`parameter-ARN`* es el ARN del parámetro de Systems Manager.

En el ejemplo siguiente, se crea una plantilla de lanzamiento que obtiene el ID de AMI de un parámetro de Systems Manager existente con el ARN de *`arn:aws:ssm:us-east-2:123456789012:parameter/MyParameter`*.

```
aws ec2 create-launch-template --launch-template-name my-template-for-auto-scaling
 --version-description version1 \
  --launch-template-data file://config.json
```

Contenido de `config.json`:

```
{
    "ImageId":"resolve:ssm:arn:aws:ssm:us-east-2:123456789012:parameter/
MyParameter",
    "InstanceType":"t2.micro"
}
```

La versión predeterminada del parámetro, cuando no se especifica, es la versión más reciente.

El siguiente ejemplo hace referencia a una versión específica del parámetro *`MyParameter`*. El ejemplo usa la versión *`3`* del parámetro *`MyParameter`*, pero puede usar cualquier número de versión válido.

```
{
    "ImageId":"resolve:ssm:arn:aws:ssm:us-east-2:123456789012:parameter/
MyParameter:3",
    "InstanceType":"t2.micro"
}
```

El siguiente ejemplo similar hace referencia a una etiqueta del parámetro *prod* que asigna a una versión específica del parámetro *MyParameter*.

```
{
    "ImageId":"resolve:ssm:arn:aws:ssm:us-east-2:123456789012:parameter/
MyParameter:prod",
    "InstanceType":"t2.micro"
}
```

A continuación, se muestra un ejemplo de respuesta.

```
{
    "LaunchTemplate": {
        "LaunchTemplateId": "lt-00f93d4588example",
        "LaunchTemplateName": "my-template-for-auto-scaling",
        "CreateTime": "2024-01-08T12:43:21.000Z",
        "CreatedBy": "arn:aws:iam::123456789012:user/Bob",
        "DefaultVersionNumber": 1,
        "LatestVersionNumber": 1
    }
}
```

Para especificar un parámetro del almacén de parámetros en una plantilla de lanzamiento, debe tener el permiso `ssm:GetParameters` para el parámetro especificado. Cualquier persona que utilice la plantilla de lanzamiento también necesitará el permiso `ssm:GetParameters` para validar el valor del parámetro. Para más información, consulte Restricción del acceso a los parámetros de Systems Manager mediante políticas de IAM en la Guía del usuario de AWS Systems Manager .

Verificación de que la plantilla de lanzamiento tenga el ID de AMI correcto

Utilice el comando describe-launch-template-versions e incluya la opción `--resolve-alias` para resolver el parámetro con el ID de AMI real.

```
aws ec2 describe-launch-template-versions --launch-template-name my-template-for-auto-
scaling \
   --versions 1 --resolve-alias
```

El ejemplo devuelve el ID de AMI de `ImageId`. Cuando se lanza una instancia con esta plantilla de lanzamiento, el ID de AMI se resuelve con `ami-0ac394d6a3example`.

```
{
    "LaunchTemplateVersions": [
        {
            "LaunchTemplateId": "lt-089c023a30example",
            "LaunchTemplateName": "my-template-for-auto-scaling",
            "VersionNumber": 1,
            "CreateTime": "2022-12-28T19:52:27.000Z",
            "CreatedBy": "arn:aws:iam::123456789012:user/Bob",
            "DefaultVersion": true,
            "LaunchTemplateData": {
                "ImageId": "ami-0ac394d6a3example",
                "InstanceType": "t2.micro",
            }
        }
    ]
}
```

Recursos relacionados

Para más información sobre cómo especificar un parámetro de Systems Manager en una plantilla de lanzamiento, consulte Uso de un parámetro de Systems Manager en lugar de un ID de AMI en la Guía del usuario de Amazon EC2.

Para obtener más información sobre cómo trabajar con parámetros de Systems Manager, consulte los siguientes materiales de referencia en la documentación de Systems Manager.

- Para crear versiones y etiquetas de parámetros, consulte Trabajo con versiones de parámetros y Trabajo con etiquetas de parámetros.
- Para obtener información sobre cómo buscar los parámetros públicos de la AMI compatibles con Amazon EC2, consulte Calling AMI public parameters.
- Para obtener información sobre cómo compartir parámetros con otras AWS cuentas o a través de ellas AWS Organizations, consulte Trabajar con parámetros compartidos.
- Para obtener información sobre la supervisión para controlar si los parámetros se crearon correctamente, consulte Native parameter support for Amazon Machine Image IDs.

Limitaciones

Cuando trabaje con parámetros de Systems Manager, tenga en cuenta las siguientes limitaciones:

- Amazon EC2 Auto Scaling solo admite la especificación de los ID de AMI como parámetros.
- No se admite la creación o actualización de grupos de instancias mixtos con una selección del tipo de instancia basada en atributos mediante una plantilla de lanzamiento que especifique un parámetro de Systems Manager.
- Si su grupo de escalado automático usa una plantilla de lanzamiento que especifique un parámetro de Systems Manager, no podrá iniciar una actualización de instancias con la configuración deseada o utilizando la coincidencia por omisión.
- Si su grupo de escalado automático usa una plantilla de lanzamiento que especifique un parámetro de Systems Manager, no se admitirán los grupos en caliente.
- En cada llamada para crear o actualizar el grupo de escalado automático, Amazon EC2 Auto Scaling resolverá el parámetro de Systems Manager de la plantilla de lanzamiento. Si utiliza parámetros avanzados o límites de rendimiento más altos, las llamadas frecuentes al Almacén de parámetros (es decir, la operación `GetParameters`) pueden aumentar los costos de Systems Manager, ya que se cobran cargos por interacción con la API del Almacén de parámetros. Para obtener más información, consulte Precios de AWS Systems Manager.

Configuraciones de inicialización de Auto Scaling

Important

Limitaciones:

- Desde el 1 de enero de 2023, los nuevos tipos de instancias de Amazon EC2 ya no son compatibles con las configuraciones de lanzamiento. Esto incluye la compatibilidad con cualquier tipo de instancia que se añada a una Región de AWS después del lanzamiento inicial de la región.
- Las cuentas creadas el 1 de junio de 2023, o después de dicha fecha, no pueden crear nuevas configuraciones de lanzamiento con la consola.
- Las cuentas creadas a partir del 1 de octubre de 2024 no pueden crear nuevas configuraciones de lanzamiento mediante ningún método (consola AWS CLI, API o CloudFormation).

Migre a plantillas de inicialización para asegurarse de no tener que crear nuevas configuraciones de inicialización ahora o en el futuro. Para obtener información sobre la migración de sus grupos de escalado automático a plantillas de lanzamiento, consulte Migración de sus grupos de escalado automático a plantillas de lanzamiento.

Note

Es posible que se pueda crear una configuración de lanzamiento con un tipo de instancia que ya no sea compatible en una región. Se recomienda que migre a las plantillas de lanzamiento.

Proporcionamos información sobre las configuraciones de lanzamiento para los clientes que aún no han migrado las configuraciones de lanzamiento a las plantillas de lanzamiento. Una configuración de lanzamiento es una plantilla de configuración de instancia que utiliza un grupo de Auto Scaling para lanzar instancias EC2. Cuando se crea una configuración de lanzamiento, se especifica información sobre las instancias. Incluya el ID de la Amazon Machine Image (AMI), el tipo de instancia, un par de claves, uno o varios grupos de seguridad y una asignación de dispositivos de

bloques. Si ha lanzado una instancia EC2 con anterioridad, habrá especificado la misma información para lanzar la instancia.

Puede especificar la configuración de lanzamiento con varios grupos de Auto Scaling. Sin embargo, solo puede especificar una configuración de lanzamiento para un grupo de escalado automático cada vez y no puede modificar una configuración de lanzamiento una vez creada. Para cambiar la configuración de lanzamiento de un grupo de escalado automático, debe crear una configuración de lanzamiento y, a continuación, actualizar el grupo de escalado automático con ella.

Contenido

- Crear una configuración de lanzamiento
- Cambio en la configuración de lanzamiento de un grupo de escalado automático

Crear una configuración de lanzamiento

Important

Limitaciones:

- Desde el 1 de enero de 2023, los nuevos tipos de instancias de Amazon EC2 ya no son compatibles con las configuraciones de lanzamiento. Esto incluye la compatibilidad con cualquier tipo de instancia que se añada Región de AWS después del lanzamiento inicial de la región.
- Las cuentas creadas el 1 de junio de 2023, o después de dicha fecha, no pueden crear nuevas configuraciones de lanzamiento con la consola.
- Las cuentas creadas a partir del 1 de octubre de 2024 no pueden crear nuevas configuraciones de lanzamiento mediante ningún método (consola AWS CLI, API o CloudFormation).

Migre a plantillas de inicialización para asegurarse de no tener que crear nuevas configuraciones de inicialización ahora o en el futuro. Para obtener información sobre la migración de sus grupos de escalado automático a plantillas de lanzamiento, consulte Migración de sus grupos de escalado automático a plantillas de lanzamiento.

> **Note**
>
> Es posible que se pueda crear una configuración de lanzamiento con un tipo de instancia que ya no sea compatible en una región. Se recomienda que migre a las plantillas de lanzamiento.

En este tema, se describe cómo crear una configuración de inicialización. Proporcionamos información sobre las configuraciones de lanzamiento para los clientes que aún no han migrado las configuraciones de lanzamiento a las plantillas de lanzamiento.

No se puede modificar una configuración de inicialización después de crearla. En su lugar, debe crear una nueva configuración de inicialización.

Para asociar una nueva configuración de inicialización a un grupo de escalado automático existente, consulte Cambio en la configuración de lanzamiento de un grupo de escalado automático. Para crear un nuevo grupo de escalado automático, consulte Crear un grupo de Auto Scaling mediante una configuración de lanzamiento.

Contenido

- Crear una configuración de lanzamiento
- Configurar las opciones de metadatos de instancia
- Crear una configuración de lanzamiento con una instancia EC2

Crear una configuración de lanzamiento

Para crear una configuración de lanzamiento utilizando la (consola)

1. Abra la consola de Amazon EC2 en https://console.aws.amazon.com/ec2/.
2. En la barra de navegación superior, selecciona tu AWS región.
3. En el panel de navegación izquierdo, en Escalado automático, elija Grupos de escalado automático.
4. Elija Configuraciones de lanzamiento cerca de la parte superior de la página. Cuando se le pida confirmación, elija Ver configuraciones de lanzamiento para confirmar que desea ver la página Configuraciones de lanzamiento.

5. Elija Create launch configuration (Crear una configuración de lanzamiento) e ingrese un nombre para la configuración de lanzamiento.
6. Para Amazon machine image (AMI), elija una AMI. Para encontrar una AMI específica, puede buscar una AMI adecuada, anote su ID e ingrese el ID como criterio de búsqueda.

   Para obtener el ID de la AMI de Amazon Linux 2:

   a. Abra la consola de Amazon EC2 en https://console.aws.amazon.com/ec2/.
   b. En el panel de navegación izquierdo, en Instancias, elija Instancias y luego elija Lanzar instancias.
   c. En la pestaña Quick Start (Inicio rápido) de la página Choose an Amazon Machine Image (Elegir una Amazon Machine Image), observe el ID de la AMI junto a Amazon Linux 2 AMI (HVM) (AMI de Amazon Linux 2 A [HVM]).
7. Para Instance Type (Tipo de instancias), seleccione la configuración de hardware de la instancia.
8. En Additional configuration (Configuración adicional), para Advanced details (Detalles avanzados), IP address type (Tipo de dirección IP), haga una selección:

   a. (Opcional) Para Purchasing option (Opción de compra), puede elegir Request Spot Instances (Solicitar instancias de spot) para solicitar instancias de spot al precio de spot, limitado al precio bajo demanda. Si lo prefiere, puede especificar un precio máximo por hora de instancia para las instancias de spot.

      > **Note**
      >
      > Las instancias de spot son una opción rentable en comparación con las instancias bajo demanda, si es flexible con respecto a cuándo es necesario ejecutar sus aplicaciones y si sus aplicaciones se pueden interrumpir. Para obtener más información, consulte Solicitud de instancias de spot para aplicaciones flexibles y tolerantes a errores.

   b. (Opcional) En IAM instance profile (Perfil de instancias de IAM), elija el rol que desea asociar a las instancias. Para obtener más información, consulte Rol de IAM para aplicaciones que se ejecuten en instancias de Amazon EC2.
   c. (Opcional) Para la monitorización, elige si quieres permitir que las instancias publiquen datos de métricas a intervalos de 1 minuto en Amazon CloudWatch habilitando la monitorización detallada. Se aplican cargos adicionales de . Para obtener más información, consulte Configuración de la supervisión para instancias de Auto Scaling.

d. (Opcional) Para Advanced details (Detalles avanzados), User data (Datos de usuario), puede especificar los datos de usuario para configurar una instancia durante el lanzamiento o para ejecutar un script de configuración después de que se lance la instancia.

e. (Opcional) Para Advanced details (Detalles avanzados), IP address type (Tipo de dirección IP), elija si desea asignar una dirección IP pública a las instancias del grupo. Si no establece ningún valor, el valor predeterminado es utilizar la configuración de IP pública de asignación automática de las subredes en las que se lanzan las instancias.

9. (Opcional) Para Storage (volumes) Almacenamiento [volúmenes], si no necesita almacenamiento adicional, puede omitir esta sección. De lo contrario, para especificar los volúmenes que desea adjuntar a las instancias además de los volúmenes especificados por la AMI, elija Add new volume (Agregar nuevo volumen). A continuación, elija las opciones deseadas y los valores asociados para Devices (Dispositivos), Snapshot (Instantánea), Size (Tamaño), Volume type (Tipo de volumen), IOPS,Throughput (Rendimiento), Delete on termination (Eliminar al terminar) y Encrypted (cifrado).

10. Para Security groups (grupos de seguridad), cree o seleccione el grupo de seguridad que se va a asociar con las instancias del grupo. Si deja la opción Create a new security group (Crear un nuevo grupo de seguridad) seleccionada como predeterminada, se configura una regla de SSH para instancias de Amazon EC2 que ejecutan Linux. Se configura una regla de RDP predeterminada para instancias de Amazon EC2 que ejecutan Windows.

11. Para Key pair (login) (Par de claves [inicio de sesión]), elija una opción en Key pair options (Opciones de par de claves).

    Si ya ha configurado un par de claves de la instancia de Amazon EC2, puede elegirlo aquí.

    Si aún no tiene un par de claves de instancia de Amazon EC2, elija Create a new key pair (Crear un nuevo par de claves) y asígnele un nombre fácil de reconocer. Elija Download Key Pair (Descargar par de claves) para descargar el par de claves en su equipo.

    > **Important**
    >
    > No elija Proceed without a key pair (Continuar si un par de claves) si necesita establecer conexión con las instancias.

12. Seleccione la casilla de confirmación y, a continuación, elija Create launch configuration.

Cómo crear una configuración de inicialización desde una configuración de inicialización existente (consola)

1. Abra la consola de Amazon EC2 en https://console.aws.amazon.com/ec2/.
2. En la barra de navegación superior, selecciona tu AWS región.
3. En el panel de navegación izquierdo, en Escalado automático, elija Grupos de escalado automático.
4. Elija Configuraciones de lanzamiento cerca de la parte superior de la página. Cuando se le pida confirmación, elija Ver configuraciones de lanzamiento para confirmar que desea ver la página Configuraciones de lanzamiento.
5. Seleccione la configuración de lanzamiento y elija Actions, Copy launch configuration. Se definirá una nueva configuración de lanzamiento con las mismas opciones que la original, pero con el texto "Copy" añadido al nombre.
6. En la página Copy Launch Configuration, modifique las opciones de configuración según sea necesario y seleccione Create launch configuration.

Para crear una configuración de lanzamiento mediante la línea de comandos

Puede utilizar uno de los siguientes comandos:

- create-launch-configuration (AWS CLI)
- Nueva ASLaunch configuración (AWS Tools for Windows PowerShell)

Configurar las opciones de metadatos de instancia

Amazon EC2 Auto Scaling admite la configuración del Servicio de metadatos de instancia (IMDS) en configuraciones de lanzamiento. Esto le da la opción de usar configuraciones de lanzamiento para configurar las instancias de Amazon EC2 en sus grupos de Auto Scaling para que requieran la versión 2 (IMDSv2) del Servicio de metadatos de instancias, que es un método orientado a la sesión para solicitar metadatos de instancias. Para obtener más información sobre IMDSv2 sus ventajas, consulte este artículo del AWS blog sobre las mejoras para añadir una defensa exhaustiva al servicio de metadatos de instancias EC2.

Puede configurar el IMDS para que sea compatible con IMDSv2 y IMDSv1 (el predeterminado) o para que requiera el uso de. IMDSv2 Si utiliza el IMDS AWS CLI o uno de ellos SDKs para configurar el IMDS, debe utilizar la última versión del mismo AWS CLI o el SDK para requerir el uso de. IMDSv2

Puede establecer la configuración de lanzamiento para lo siguiente:

- Exija el uso de IMDSv2 cuando solicite los metadatos de la instancia
- Especificar el límite de saltos de respuesta de PUT
- Desactivar el acceso a los metadatos de instancia

Puede encontrar más información sobre la configuración del Servicio de metadatos de instancia en el siguiente tema: Configuración del servicio de metadatos de instancia en la Guía del usuario de Amazon EC2.

Utilice el siguiente procedimiento para configurar las opciones IMDS en una configuración de lanzamiento. Después de crear la configuración de lanzamiento, puede asociarla a su grupo de escalado automático. Si asocia la configuración de lanzamiento con un grupo de escalado automático existente, la configuración de lanzamiento existente se desasocia del grupo Auto Scaling y las instancias existentes requerirán reemplazo para utilizar las opciones IMDS especificadas en la nueva configuración de lanzamiento. Para obtener más información, consulte Cambio en la configuración de lanzamiento de un grupo de escalado automático.

Para configurar IMDS en una configuración de lanzamiento (consola)

1. Abra la consola de Amazon EC2 en https://console.aws.amazon.com/ec2/.
2. En la barra de navegación superior, selecciona tu AWS región.
3. En el panel de navegación izquierdo, en Escalado automático, elija Grupos de escalado automático.
4. Elija Configuraciones de lanzamiento cerca de la parte superior de la página. Cuando se le pida confirmación, elija Ver configuraciones de lanzamiento para confirmar que desea ver la página Configuraciones de lanzamiento.
5. Seleccione Create launch configuration (Crear una configuración de lanzamiento) y cree la configuración de lanzamiento de la forma habitual. Incluya el ID de la Amazon Machine Image(AMI), el tipo de instancias y, de forma opcional, un par de claves, uno o varios grupos de seguridad y cualquier volumen de EBS o de almacenamiento de instancias adicionales para sus instancias.
6. Para configurar las opciones de metadatos de instancia para todas las instancias asociadas a esta configuración de lanzamiento, en Additional configuration (Configuración adicional), en Advanced details (Detalles avanzados), realice una de las siguientes opciones:

a. Para Metadata accessible (Metadatos accesibles), elija si habilitar o deshabilitar el acceso al punto de conexión HTTP del servicio de metadatos de instancia. De forma predeterminada, el punto de conexión HTTP está habilitado. Si decide desactivar el punto de conexión, el acceso a los metadatos de la instancia está desactivado. Puede especificar la condición que se va a requerir IMDSv2 solo cuando el punto final HTTP esté habilitado.

b. Para la versión de metadatos, puede optar por exigir el uso de la versión 2 (IMDSv2) del servicio de metadatos de la instancia al solicitar los metadatos de la instancia. Si no especificas ningún valor, el valor predeterminado es admitir ambos IMDSv1 valores IMDSv2.

c. Para Metadata token response hop limit (Límite de saltos de respuesta del token de metadatos), puede establecer el número permitido de saltos de red para el token de metadatos. Si no especifica un valor, el predeterminado es 1.

7. Cuando haya terminado, seleccione Create launch configuration (Crear una configuración de lanzamiento).

Para requerir el uso de IMDSv2 en una configuración de lanzamiento mediante el AWS CLI

Utilice el siguiente create-launch-configurationcomando con el `--metadata-options` valor establecido en`HttpTokens=required`. Cuando se especifica un valor para `HttpTokens`, también se debe establecer en `HttpEndpoint` para habilitarlo. Como el encabezado del token seguro está configurado como obligatorio para las solicitudes de recuperación de metadatos, en la instancia se opta por requerir su uso IMDSv2 al solicitar los metadatos de la instancia.

```
aws autoscaling create-launch-configuration \
  --launch-configuration-name my-lc-with-imdsv2 \
  --image-id ami-01e24be29428c15b2 \
  --instance-type t2.micro \
      ...
  --metadata-options "HttpEndpoint=enabled,HttpTokens=required"
```

Para desactivar el acceso a los metadatos de instancia

Usa el siguiente create-launch-configurationcomando para desactivar el acceso a los metadatos de la instancia. Puedes volver a activar el acceso más adelante mediante el modify-instance-metadata-optionscomando.

```
aws autoscaling create-launch-configuration \
```

```
--launch-configuration-name my-lc-with-imds-disabled \
--image-id ami-01e24be29428c15b2 \
--instance-type t2.micro \
    ...
--metadata-options "HttpEndpoint=disabled"
```

Crear una configuración de lanzamiento con una instancia EC2

También tiene la opción de crear una configuración de inicialización mediante los atributos de una instancia de EC2 en ejecución.

Existen diferencias entre crear una configuración de lanzamiento desde cero y crear una configuración de lanzamiento desde una instancia EC2 existente. Cuando crea una configuración de lanzamiento desde cero, debe especificar el ID de la imagen, el tipo de instancia, los recursos opcionales (como dispositivos de almacenamiento) y ajustes opcionales (como la monitorización). Cuando crea una configuración de lanzamiento desde una instancia en ejecución, Amazon EC2 Auto Scaling obtiene los atributos de la configuración de lanzamiento de la instancia especificada. Los atributos también se derivan del mapeo de dispositivos de bloques de la AMI desde la que se lanzó la instancia, omitiendo los dispositivos de bloques adicionales que se añadieron tras el lanzamiento.

Si crea una configuración de lanzamiento con una instancia en ejecución, puede invalidar los siguientes atributos especificándolos como parte de la misma solicitud: AMI, dispositivos de bloques, par de claves, perfil de instancias, tipo de instancias, kernel, monitoreo de instancias, tenencia de ubicación, disco RAM, grupos de seguridad, precio de spot (máximo), datos del usuario, si la instancia tiene una dirección IP pública y si la instancia está optimizada para EBS.

> **Note**
>
> Si la instancia especificada tiene propiedades que no admiten actualmente las configuraciones de lanzamiento, las instancias lanzadas por el grupo de escalado automático puede que no sean idénticas a la instancia EC2 original.

> **Important**
>
> La AMI que se utiliza para lanzar la instancia especificada debe existir.

Temas

- Crear una configuración de lanzamiento desde una instancia EC2 (AWS CLI)
- Crear una configuración de lanzamiento desde una instancia e invalidar los dispositivos de bloques (AWS CLI)
- Crear una configuración de lanzamiento e invalidar el tipo de instancias (AWS CLI)

Crear una configuración de lanzamiento desde una instancia EC2 (AWS CLI)

Utilice el siguiente comando create-launch-configuration para crear una configuración de lanzamiento desde una instancia con los mismos atributos que la instancia. Todos los dispositivos de bloques añadidos después del lanzamiento se omiten.

```
aws autoscaling create-launch-configuration --launch-configuration-name my-lc-from-instance --instance-id i-a8e09d9c
```

Puedes usar el siguiente describe-launch-configurationscomando para describir la configuración de lanzamiento y comprobar que sus atributos coinciden con los de la instancia.

```
aws autoscaling describe-launch-configurations --launch-configuration-names my-lc-from-instance
```

A continuación, se muestra un ejemplo de respuesta.

```
{
    "LaunchConfigurations": [
        {
            "UserData": null,
            "EbsOptimized": false,
            "LaunchConfigurationARN": "arn",
            "InstanceMonitoring": {
                "Enabled": false
            },
            "ImageId": "ami-05355a6c",
            "CreatedTime": "2014-12-29T16:14:50.382Z",
            "BlockDeviceMappings": [],
            "KeyName": "my-key-pair",
            "SecurityGroups": [
                "sg-8422d1eb"
            ],
            "LaunchConfigurationName": "my-lc-from-instance",
```

```
            "KernelId": "null",
            "RamdiskId": null,
            "InstanceType": "t1.micro",
            "AssociatePublicIpAddress": true
        }
    ]
}
```

Crear una configuración de lanzamiento desde una instancia e invalidar los dispositivos de bloques (AWS CLI)

De forma predeterminada, Amazon EC2 Auto Scaling utiliza los atributos de la instancia EC2 que especifica para crear la configuración de lanzamiento. Sin embargo, los dispositivos de bloques proceden de la AMI usada para lanzar la instancia, no de la instancia. Para añadir dispositivos de bloques a la configuración de lanzamiento, invalide el mapeo de dispositivos de bloques para la configuración de lanzamiento.

Utilice el siguiente create-launch-configurationcomando para crear una configuración de lanzamiento mediante una instancia EC2 pero con un mapeo de dispositivos de bloques personalizado.

```
aws autoscaling create-launch-configuration --launch-configuration-name my-lc-from-instance-bdm --instance-id i-a8e09d9c \
  --block-device-mappings "[{\"DeviceName\":\"/dev/sda1\",\"Ebs\":{\"SnapshotId\":\"snap-3decf207\"}},{\"DeviceName\":\"/dev/sdf\",\"Ebs\":{\"SnapshotId\":\"snap-eed6ac86\"}}]"
```

Utilice el siguiente describe-launch-configurationscomando para describir la configuración de lanzamiento y comprobar que utiliza su mapeo de dispositivos de bloques personalizado.

```
aws autoscaling describe-launch-configurations --launch-configuration-names my-lc-from-instance-bdm
```

En la siguiente respuesta de ejemplo se describe la configuración de lanzamiento.

```
{
    "LaunchConfigurations": [
        {
            "UserData": null,
            "EbsOptimized": false,
            "LaunchConfigurationARN": "arn",
```

```
            "InstanceMonitoring": {
                "Enabled": false
            },
            "ImageId": "ami-c49c0dac",
            "CreatedTime": "2015-01-07T14:51:26.065Z",
            "BlockDeviceMappings": [
                {
                    "DeviceName": "/dev/sda1",
                    "Ebs": {
                        "SnapshotId": "snap-3decf207"
                    }
                },
                {
                    "DeviceName": "/dev/sdf",
                    "Ebs": {
                        "SnapshotId": "snap-eed6ac86"
                    }
                }
            ],
            "KeyName": "my-key-pair",
            "SecurityGroups": [
                "sg-8637d3e3"
            ],
            "LaunchConfigurationName": "my-lc-from-instance-bdm",
            "KernelId": null,
            "RamdiskId": null,
            "InstanceType": "t1.micro",
            "AssociatePublicIpAddress": true
        }
    ]
}
```

Crear una configuración de lanzamiento e invalidar el tipo de instancias (AWS CLI)

De forma predeterminada, Amazon EC2 Auto Scaling utiliza los atributos de la instancia EC2 que se especifican para crear la configuración de lanzamiento. En función de sus requisitos, tal vez desee invalidar los atributos de la instancia y utilizar los valores que necesite. Por ejemplo, puede invalidar el tipo de instancia.

Utilice el siguiente create-launch-configurationcomando para crear una configuración de lanzamiento con una instancia EC2 pero con un tipo de instancia diferente (por ejemplo`t2.medium`) al de la instancia (por ejemplo`t2.micro`).

```
aws autoscaling create-launch-configuration --launch-configuration-name my-lc-from-instance-changetype \
  --instance-id i-a8e09d9c --instance-type t2.medium
```

Usa el siguiente describe-launch-configurationscomando para describir la configuración de lanzamiento y verificar que se haya anulado el tipo de instancia.

```
aws autoscaling describe-launch-configurations --launch-configuration-names my-lc-from-instance-changetype
```

En la siguiente respuesta de ejemplo se describe la configuración de lanzamiento.

```
{
    "LaunchConfigurations": [
        {
            "UserData": null,
            "EbsOptimized": false,
            "LaunchConfigurationARN": "arn",
            "InstanceMonitoring": {
                "Enabled": false
            },
            "ImageId": "ami-05355a6c",
            "CreatedTime": "2014-12-29T16:14:50.382Z",
            "BlockDeviceMappings": [],
            "KeyName": "my-key-pair",
            "SecurityGroups": [
                "sg-8422d1eb"
            ],
            "LaunchConfigurationName": "my-lc-from-instance-changetype",
            "KernelId": "null",
            "RamdiskId": null,
            "InstanceType": "t2.medium",
            "AssociatePublicIpAddress": true
        }
    ]
}
```

Cambio en la configuración de lanzamiento de un grupo de escalado automático

> **Important**
>
> Proporcionamos información sobre las configuraciones de lanzamiento para los clientes que aún no han migrado las configuraciones de lanzamiento a las plantillas de lanzamiento. Para obtener información sobre la migración de sus grupos de escalado automático a plantillas de lanzamiento, consulte Migración de sus grupos de escalado automático a plantillas de lanzamiento.

En este tema, se describe cómo asociar una configuración de inicialización diferente al grupo de escalado automático.

Después de cambiar la configuración de inicialización, las nuevas instancias se inicializan con las nuevas opciones de configuración, pero las instancias existentes no resultan afectadas. Para obtener más información, consulte Actualización de las instancias de escalado automático.

Para reemplazar la configuración de lanzamiento de un grupo de escalado automático (consola)

1. Abra la consola de Amazon EC2 en https://console.aws.amazon.com/ec2/.
2. En el panel de navegación izquierdo, en Escalado automático, elija Grupos de escalado automático.
3. Seleccione la casilla situada junto al grupo de escalado automático.

   Se abre un panel dividido en la parte inferior de la página.
4. En la pestaña Details (Detalles), elija Launch configurations (Configuraciones de lanzamiento), Edit (Editar).
5. En Configuración de inicialización, seleccione la nueva configuración de inicialización.
6. Elija Update (Actualizar) cuando haya terminado.

Cómo cambiar la configuración de inicialización de un grupo de escalado automático con la línea de comandos

Puede utilizar uno de los siguientes comandos:

- update-auto-scaling-group (AWS CLI)
- Actualizar- () ASAuto ScalingGroupAWS Tools for Windows PowerShell

Grupos de escalado automático

> **Note**
>
> Si es la primera vez que usa los grupos de escalado automático, siga los pasos del tutorial Cómo crear su primer grupo de escalado automático para empezar y ver cómo responde un grupo de escalado automático cuando finaliza una instancia del grupo.

Un Auto Scaling group (grupo de escalado automático) contiene una colección de instancias de Amazon EC2 que se tratan como una agrupación lógica a efectos de escalado automático y administración. Un grupo de escalado automático también le permite utilizar características de Amazon EC2 Auto Scaling, como sustituciones de comprobaciones de estado y políticas de escalado. Tanto el mantenimiento del número de instancias en un grupo de Auto Scaling como el escalado automático son las funcionalidades básicas del servicio de Amazon EC2 Auto Scaling.

El tamaño de un grupo de Auto Scaling depende del número de instancias que establezca como capacidad deseada. Puede ajustar su tamaño para satisfacer la demanda, ya sea de forma manual o mediante el uso de escalado automático.

Un grupo de Auto Scaling comienza lanzando suficientes instancias para satisfacer su capacidad deseada. Mantiene este número de instancias realizando comprobaciones de estado periódicas en las instancias del grupo. El grupo de Auto Scaling sigue manteniendo un número fijo de instancias aunque una de ellas deje de estar en buen estado. Si una instancia pasa a tener un estado incorrecto, el grupo termina la instancia en mal estado y lanza otra instancia para sustituirla. Para obtener más información, consulte Comprobaciones de estado para instancias en un grupo de escalado automático.

Puede utilizar políticas de escalado para aumentar o disminuir dinámicamente el número de instancias de su grupo para satisfacer las condiciones cambiantes. Cuando la política de escalado entra en vigor, el grupo de Auto Scaling ajusta la capacidad deseada del grupo, entre los valores de capacidad mínima y máxima, y lanza o termina las instancias según sea necesario. También puede efectuar el escalado mediante una programación. Para obtener más información, consulte Elección de su método de escalado.

Al crear un grupo de Auto Scaling, puede elegir si desea lanzar On-Demand instancias, instancias puntuales o ambas. Puede especificar varias opciones de compra para el grupo de escalado

automático solo cuando configure el grupo para que utilice una launch template. Para obtener más información, consulte Grupos de Auto Scaling con varios tipos de instancia y opciones de compra.

Las instancias puntuales le proporcionan acceso a la capacidad de EC2 no utilizada con grandes descuentos en relación con los On-Demand precios. Para obtener más información, consulte Precios de instancias de spot de Amazon EC2. Existen diferencias clave entre las instancias puntuales y las On-Demand instancias:

- El precio de las instancias de spot varía según la demanda
- Amazon EC2 puede terminar una instancia de spot individual a medida que cambie la disponibilidad o el precio de las instancias de spot

Cuando se termina una instancia de spot, el grupo de Auto Scaling intenta lanzar una instancia de sustitución para mantener la capacidad deseada para el grupo.

Cuando las instancias se lancen, si especifica varias zonas de disponibilidad, la capacidad deseada se distribuye entre ellas. Si se produce una acción de escalado, Amazon EC2 Auto Scaling mantiene automáticamente el equilibrio en todas las zonas de disponibilidad especificadas.

Contenido

- Creación de grupos de escalado automático mediante plantillas de lanzamiento
- Creación de grupos de escalado automático mediante configuraciones de lanzamiento
- Lanza instancias de forma sincrónica
- Update an Auto Scaling group
- Etiquetado de grupos e instancias de Auto Scaling
- Políticas de mantenimiento de instancias
- Enlaces de ciclo de vida de Amazon EC2 Auto Scaling
- Reducción de la latencia de las aplicaciones con tiempos de arranque prolongados mediante grupos de calentamiento
- Cambio zonal del grupo Auto Scaling
- Distribución de zonas de disponibilidad del grupo de escalado automático
- Desconexión o asociación de instancias en el grupo de escalado automático
- Eliminación temporal de las instancias de un grupo de escalado automático
- Eliminación de la infraestructura de Auto Scaling

Creación de grupos de escalado automático mediante plantillas de lanzamiento

Si ha creado una plantilla de lanzamiento, puede crear un grupo de escalado automático que use una plantilla de lanzamiento como plantilla de configuración para sus instancias de EC2. La plantilla de lanzamiento especifica información como el ID de AMI, el tipo de instancia, el par de claves, los grupos de seguridad y la asignación de dispositivos de bloques para las instancias. Para obtener información acerca de la creación de plantillas de lanzamiento, consulte Creación de una plantilla de lanzamiento para un grupo de Auto Scaling.

Usted debe tener los permisos suficientes para poder crear un grupo de escalado automático. También debe tener permisos suficientes para crear el rol vinculado a servicio que Amazon EC2 Auto Scaling utiliza para realizar acciones en su nombre si este no existe todavía. Para ver ejemplos de políticas de IAM que un administrador puede utilizar como referencia para concederle permisos, consulte Ejemplos de políticas basadas en identidades y Control del uso de la plantilla de inicialización de Amazon EC2 en los grupos de escalado automático.

Contenido

- Creación de un grupo de Auto Scaling mediante una plantilla de lanzamiento
- Creación de un grupo de Auto Scaling mediante el asistente de lanzamiento de Amazon EC2
- Grupos de Auto Scaling con varios tipos de instancia y opciones de compra

Creación de un grupo de Auto Scaling mediante una plantilla de lanzamiento

Cuando crea un grupo de Auto Scaling, debe especificar la información necesaria para configurar las instancias de Amazon EC2, las zonas de disponibilidad y las subredes de VPC para las instancias, la capacidad deseada y los límites de capacidad mínima y máxima.

Para configurar instancias de Amazon EC2 lanzadas por el grupo de Auto Scaling, puede especificar una plantilla de lanzamiento o una configuración de lanzamiento. El siguiente procedimiento demuestra cómo crear un grupo de Auto Scaling mediante una plantilla de lanzamiento.

Requisitos previos

- Tiene que haber creado una plantilla de lanzamiento. Para obtener más información, consulte Creación de una plantilla de lanzamiento para un grupo de Auto Scaling.

Para crear un grupo de Auto Scaling mediante una plantilla de lanzamiento (consola)

1. Abra la consola Amazon EC2 en https://console.aws.amazon.com/ec2/y seleccione Auto Scaling Groups en el panel de navegación.
2. En la barra de navegación situada en la parte superior de la pantalla, elija la misma Región de AWS que utilizó al crear la plantilla de lanzamiento.
3. Elija Create an Auto Scaling group (Crear un grupo de escalado automático).
4. En la página Choose launch template or configuration (Elegir plantilla o configuración de lanzamiento) haga lo siguiente:

   a. Para Nombre de grupo de Auto Scaling, ingrese un nombre para el grupo de Auto Scaling.

   b. En launch template (Plantilla de lanzamiento), elija una plantilla de lanzamiento existente.

   c. Para Launch template version (Versión de plantilla de lanzamiento), decida si el grupo de escalado automático utiliza el valor predeterminado, la última versión o una versión específica de la plantilla de lanzamiento para escalado horizontal.

   d. Compruebe que la plantilla de lanzamiento admita todas las opciones que tiene previsto utilizar y, a continuación, elija Next (Siguiente).
5. En la página Elija opciones de lanzamiento de la instancia, si no utiliza varios tipos de instancias, puede omitir la sección Requisitos del tipo de instancia para usar el tipo de instancia de EC2 que se especifica en la plantilla de lanzamiento.

   Para usar varios tipos de instancia, consulte Grupos de Auto Scaling con varios tipos de instancia y opciones de compra.
6. En Network (Red), para la opción VPC, elija una VPC. El grupo de Auto Scaling debe crearse en la misma VPC que el grupo de seguridad especificado en la plantilla de lanzamiento.
7. En Zonas de disponibilidad y subredes, elija una o más subredes de la VPC especificada. Utilice subredes en varias zonas de disponibilidad para lograr una alta disponibilidad. Para obtener más información, consulte Consideraciones a la hora de elegir subredes de VPC.
8. Para la Distribución por zonas de disponibilidad, seleccione una estrategia de distribución. Para obtener más información, consulte Distribución de zonas de disponibilidad del grupo de escalado automático.
9. Si creó una plantilla de lanzamiento con un tipo de instancia especificado, puede continuar con el siguiente paso para crear un grupo de Auto Scaling que utilice el tipo de instancia en la plantilla de lanzamiento.

De forma alternativa, puede elegir la opción Anulación de la plantilla de inicialización si no se especifica ningún tipo de instancia en la plantilla de lanzamiento o si desea utilizar varios tipos de instancias para el escalado automático. Para obtener más información, consulte Grupos de Auto Scaling con varios tipos de instancia y opciones de compra.

10. Elija Next (Siguiente) para continuar con el siguiente paso.

    O puede aceptar el resto de las opciones predeterminadas y elegir Skip to review (Omitir para revisar).

11. (Opcional) En la página Integrar con otros servicios, configure las siguientes opciones y luego seleccione Siguiente:

    a. Para el Equilibrador de carga, elija si desea adjuntar su grupo de escalado automático a un equilibrador de carga. Para obtener más información, consulte Elastic Load Balancing.

    b. En Opciones de integración de VPC Lattice, elija si desea utilizar VPC Lattice. Para obtener más información, consulte Administración del flujo de tráfico con un grupo de destino de VPC Lattice.

    c. Para el Cambio de zona de Amazon Application Recovery Controller (ARC), seleccione la casilla de verificación para habilitar el cambio de zona. Para obtener más información, consulte Cambio zonal del grupo Auto Scaling.

       - Si habilita el cambio de zona, en Comportamiento de la comprobación de estado, seleccione Ignorar instancia en mal estado o Reemplazar instancia en mal estado. Para obtener más información, consulte Cómo funciona el cambio zonal para los grupos de Auto Scaling.

    d. En Comprobaciones de estado, Tipos de comprobaciones de estado adicionales, seleccione Activar las comprobaciones de estado de Amazon EBS. Para obtener más información, consulte Supervisión de las instancias de escalado automático con volúmenes de Amazon EBS dañados mediante comprobaciones de estado.

    e. En Período de gracia de comprobación de estado, ingrese el tiempo, en segundos. Este es la cantidad de tiempo que Amazon EC2 Auto Scaling debe esperar antes de comprobar el estado de una instancia una vez que pasa al estado `InService`. Para obtener más información, consulte Establezca el periodo de gracia de la comprobación de estado para un grupo de escalado automático.

12. (Opcional) En la página Configurar tamaño de grupo y escala, configure las siguientes opciones y, a continuación, elija Siguiente:

a. En Tamaño de grupo, para Capacidad deseada, introduzca el número inicial de instancias que desea inicializar.

b. En la sección Escalado, en Límites de escalado, si el nuevo valor de Capacidad deseada es superior a Capacidad deseada mínima y a Capacidad deseada máxima, Capacidad deseada máxima se incrementará automáticamente al nuevo valor de capacidad deseada. Puede cambiar estos límites según sea necesario. Para obtener más información, consulte Establecimiento de límites de escalado para el grupo de escalado automático.

c. En Escalado automático, elija si desea crear una política de escalado de seguimiento de destino. También puede crear esta política después de crear su grupo de escalado automático.

   Si elige Política de escalado de seguimiento de destino, siga las instrucciones en Creación de una política de escalado de seguimiento de destino para crear la política.

d. En Política de mantenimiento de instancias, elija si desea crear una política de mantenimiento de instancias. También puede crear esta política después de crear su grupo de escalado automático. Siga las instrucciones de Establecimiento de una política de mantenimiento de instancias para crear la política.

e. En Configuración de capacidad adicional, en Preferencia de reserva de capacidad, elija si desea utilizar una preferencia de reserva de capacidad. Para obtener más información, consulte Capacidad reservada en zonas de disponibilidad específicas con reserva de capacidad.

f. En Configuración adicional, Protección de reducción horizontal de instancias, elija si desea habilitar la protección de reducción horizontal de instancias. Para obtener más información, consulte Uso de la protección frente a la reducción horizontal de instancias para controlar la terminación de instancias.

g. En el caso de la supervisión, elija si desea habilitar la recopilación de métricas CloudWatch grupales. Estas métricas proporcionan mediciones que pueden ser indicadores de un posible problema, como la cantidad de instancias en proceso de terminación o la cantidad de instancias pendientes. Para obtener más información, consulte Supervise CloudWatch las métricas de sus grupos e instancias de Auto Scaling.

h. Seleccione preparación predeterminada de instancias y elija el tiempo de preparación de la aplicación. Si está creando un grupo de Auto Scaling que tiene una política de escalado, la función de calentamiento de instancias predeterminada mejora CloudWatch las métricas de Amazon utilizadas para el escalado dinámico. Para obtener más información, consulte

Establecimiento de la preparación predeterminada de instancias para un grupo de escalado automático.

13. (Opcional) En la página Agregar notificación, configure la notificación y, a continuación, elija Siguiente. Para obtener más información, consulte Opciones de notificaciones de Amazon SNS para Amazon EC2 Auto Scaling.
14. (Opcional) En la página Agregar etiquetas, seleccione Agregar etiqueta, facilite un valor y una clave para cada etiqueta, y luego seleccione Siguiente. Para obtener más información, consulte Etiquetado de grupos e instancias de Auto Scaling.
15. En la página Review (Revisar), elija Create Auto Scaling group (Crear grupo de escalado automático).

Para crear un grupo de Auto Scaling mediante la línea de comandos

Puede utilizar uno de los siguientes comandos:

- create-auto-scaling-group (AWS CLI)
- Nuevo- () ASAuto ScalingGroupAWS Tools for Windows PowerShell

Creación de un grupo de Auto Scaling mediante el asistente de lanzamiento de Amazon EC2

A continuación, se muestra el procedimiento para crear un grupo de Auto Scaling con el asistente de Launch instance (Lanzar instancia) de la consola de Amazon EC2. Con esta opción se rellena automáticamente la plantilla de lanzamiento con ciertos detalles de configuración del asistente de Launch instance (Lanzar instancia).

> **Note**
>
> El asistente no rellena el grupo de Auto Scaling con la cantidad de instancias que especifique; solo rellena la plantilla de lanzamiento con el ID de la imagen de máquina de Amazon (AMI) y el tipo de instancia. Utilice el asistente Create Auto Scaling group (Crear grupo de Auto Scaling) para especificar la cantidad de instancias que desea lanzar.
> Una AMI proporciona la información necesaria para configurar una instancia. Cuando necesite varias instancias con la misma configuración, puede lanzarlas desde una misma AMI. Recomendamos utilizar una AMI personalizada que ya tenga instalada la aplicación para evitar que las instancias terminen en caso de que reinicie una instancia que pertenece

a un grupo de Auto Scaling. Para utilizar una AMI personalizada con Amazon EC2 Auto Scaling, primero debe crear la AMI a partir de una instancia personalizada y, a continuación, utilizar la AMI para crear una plantilla de lanzamiento para su grupo de Auto Scaling.

Requisitos previos

- Debe haber creado una AMI personalizada en el mismo Región de AWS lugar donde planea crear el grupo Auto Scaling. Para obtener más información, consulte Creación de una AMI en la Guía del usuario de Amazon EC2.

Uso de una AMI personalizada como plantilla

En esta sección, usted usa el asistente de lanzamiento de Amazon EC2 para rellenar automáticamente una plantilla de lanzamiento con su AMI personalizada. Como alternativa, para configurar la plantilla de lanzamiento desde cero o para obtener una descripción más detallada de los parámetros que puede configurar para la plantilla de lanzamiento, consulte Creación de una plantilla de lanzamiento (consola).

Para utilizar una AMI personalizada como plantilla

1. Abra la consola de Amazon EC2 en https://console.aws.amazon.com/ec2/.
2. En la barra de navegación de la parte superior de la pantalla, Región de AWS se muestra la corriente. Seleccione una región para lanzar el grupo de Auto Scaling.
3. En el panel de navegación, seleccione Instances (Instancias).
4. Elija Launch instance (Lanzar instancia) y proceda del modo siguiente:
    a. En Name and tags (Nombre y etiquetas), deje Name (Nombre) en blanco. El nombre no forma parte de los datos que se utilizan para crear una plantilla de lanzamiento.
    b. En Imágenes de aplicaciones y sistemas operativos (Amazon Machine Image), elija Explorar más AMIs para ver el catálogo completo de AMI.
    c. Elija Mi AMIs, busque la AMI que creó y, a continuación, seleccione Seleccionar.
    d. En Instance type (Tipo de instancia), elija un tipo de instancia.

> **Note**
>
> Elija el mismo tipo de instancia que utilizó cuando creó la AMI o uno más potente.

e. A la derecha de la pantalla, debajo de Summary (Resumen), para Number of instances (Número de instancias), ingrese cualquier número. El número que ingrese aquí no importa. Especificará el número de instancias que desea lanzar cuando cree el grupo de Auto Scaling.

   En el campo Number of instances (Número de instancias), aparece un mensaje que dice When launching more than 1 instance, consider EC2 Auto Scaling (Al lanzar más de 1 instancia, considerar EC2 Auto Scaling).

f. Elija el texto de hipervínculo consider EC2 Auto Scaling (Considerar EC2 Auto Scaling).

g. En el diálogo de confirmación Launch into Auto Scaling Group (Lanzar en grupo de Auto Scaling), elija Continue (Continuar) para ir a la página Create launch template (Crear plantilla de lanzamiento) con la AMI y el tipo de instancia que seleccionó en el asistente de lanzamiento de instancias ya rellenado.

Después de elegir Continue (Continuar), se abre la página Create launch template (Crear plantilla de lanzamiento). Siga este procedimiento para terminar de crear una plantilla de lanzamiento.

Para crear una plantilla de lanzamiento

1. En Launch template name and description (Nombre y descripción de la plantilla de lanzamiento), ingrese un nombre y una descripción para la nueva plantilla de lanzamiento.
2. (Opcional) Debajo de Key pair (login) (Par de claves [inicio de sesión]), en Key pair name (Nombre del par de claves), elija el nombre del par de claves creado previamente que utilizará al conectarse a las instancias, por ejemplo, con SSH.
3. (Opcional) Debajo de Network settings (Configuración de red), en Security groups (Grupos de seguridad), elija uno o más grupos de seguridad creados previamente.
4. (Opcional) En Configure storage (Configurar almacenamiento), actualice la configuración del almacenamiento. La configuración de almacenamiento predeterminada está determinada por la AMI y el tipo de instancia.
5. Cuando haya terminado de configurar la plantilla de lanzamiento, elija Create launch template (Crear plantilla de lanzamiento).

6. En la página de confirmación, elija Create Auto Scaling group (Crear un grupo de Auto Scaling).

Creación de un grupo de escalado automático

> **Note**
>
> En las demás secciones del tema, se describe el procedimiento básico para crear un grupo de Auto Scaling. Para obtener una descripción más detallada de los parámetros que puede configurar para su grupo de Auto Scaling, consulte Creación de un grupo de Auto Scaling mediante una plantilla de lanzamiento.

Después de elegir Create Auto Scaling group (Crear grupo de Auto Scaling), se abre el asistente de Create Auto Scaling group (Crear grupo de Auto Scaling). Siga este procedimiento para crear un grupo de Auto Scaling.

Para crear un grupo de Auto Scaling

1. En la página Choose launch template or configuration (Elegir una plantilla de lanzamiento o configuración), escriba un nombre para el grupo de Auto Scaling.
2. La plantilla de lanzamiento que creó ya está seleccionada de forma predeterminada.

   Para Launch template version (Versión de plantilla de lanzamiento), decida si el grupo de escalado automático utiliza el valor predeterminado, la última versión o una versión específica de la plantilla de lanzamiento para escalado horizontal.
3. Elija Next (Siguiente) para continuar con el siguiente paso.
4. En la página Elija opciones de lanzamiento de la instancia, si no utiliza varios tipos de instancias, puede omitir la sección Requisitos del tipo de instancia para usar el tipo de instancia EC2 que se especifica en la plantilla de lanzamiento.

   Para usar varios tipos de instancia, consulte Grupos de Auto Scaling con varios tipos de instancia y opciones de compra.
5. En Network (Red), para la opción VPC, elija una VPC. El grupo de Auto Scaling debe crearse en la misma VPC que el grupo de seguridad especificado en la plantilla de lanzamiento.

> **Tip**
>
> Si no especificó un grupo de seguridad en la plantilla de lanzamiento, las instancias se lanzan con un grupo de seguridad predeterminado de la VPC que usted especifique. De forma predeterminada, este grupo de seguridad no permite el tráfico entrante de redes externas.

6. En Zonas de disponibilidad y subredes, elija una o más subredes de la VPC especificada.
7. Para la Distribución por zonas de disponibilidad, seleccione una estrategia de distribución. Para obtener más información, consulte Distribución de zonas de disponibilidad del grupo de escalado automático.
8. Elija Next (Siguiente) dos veces para ir a la página Configure group size and scaling policies (Configurar el tamaño del grupo y las políticas de escalado).
9. En Tamaño del grupo, defina la Capacidad deseada (cantidad inicial de instancias que se lanzarán inmediatamente después de crear el grupo de escalado automático).
10. En la sección Escalado, en Límites de escalado, si el nuevo valor de la Capacidad deseada es superior a la Capacidad deseada mínima y a la Capacidad deseada máxima, la Capacidad deseada máxima se incrementará automáticamente al nuevo valor de capacidad deseada. Puede cambiar estos límites según sea necesario. Para obtener más información, consulte Establecimiento de límites de escalado para el grupo de escalado automático.
11. Elija Skip to review (Omitir para revisar).
12. En la página Review (Revisar), elija Create Auto Scaling group (Crear grupo de escalado automático).

Siguientes pasos

Para verificar que el grupo de Auto Scaling se creó correctamente, consulte el historial de actividad. En la pestaña Activity (Actividad), en Activity history (Historial de actividad), la columna Status (Estado) muestra si su grupo de Auto Scaling lanzó las instancias correctamente. Si las instancias no se lanzan o se lanzan pero terminan inmediatamente, consulte los siguientes temas para conocer las posibles causas y soluciones:

- Solución de problemas de Amazon EC2 Auto Scaling: errores de lanzamiento de instancias de EC2
- Solución de problemas de Amazon EC2 Auto Scaling: problemas de AMI

- Solución de problemas de instancias en mal estado en Amazon EC2 Auto Scaling

Ahora puede adjuntar un equilibrador de carga en la misma región que el grupo de Auto Scaling, si lo desea. Para obtener más información, consulte Utilice Elastic Load Balancing para distribuir el tráfico entrante de la aplicación en su grupo de escalado automático..

Grupos de Auto Scaling con varios tipos de instancia y opciones de compra

Puede lanzar y escalar automáticamente una flota de instancias en diferido e instancias de spot en un solo grupo de escalado automático. Además de recibir descuentos para utilizar las instancias de spot, puede utilizar instancias reservadas o un Savings Plans para recibir descuentos en el precio regular de las instancias bajo demanda. Estos factores le permiten optimizar el ahorro de costos en las instancias de EC2, a la vez que se asegura de obtener la escala y el rendimiento deseados para su aplicación.

Las instancias de spot de son una capacidad de repuesto disponible con grandes descuentos en comparación con el precio bajo demanda de EC2. Las instancias de spot son una opción económica si es flexible con respecto a cuándo es necesario ejecutar las aplicaciones y si las aplicaciones se pueden interrumpir. Se pueden utilizar para diversas aplicaciones flexibles y tolerantes a errores. Los ejemplos incluyen servidores web sin estado, puntos finales de API, aplicaciones de macrodatos y análisis, cargas de trabajo en contenedores, CI/CD canalizaciones, computación de alto rendimiento y alto rendimiento (HPC/HTC), cargas de trabajo de renderizado y otras cargas de trabajo flexibles.

Para obtener más información, consulte Opciones de compra de instancias en la Guía del usuario de Amazon EC2.

Temas

- Información general de la configuración de creación de un grupo de instancias mixtas
- Estrategias de asignación de varios tipos de instancias
- Creación de grupos de instancias mixtas mediante la selección del tipo de instancia basada en atributos
- Creación de un grupo de instancias mixtas seleccionando manualmente los tipos de instancias
- Configuración de un grupo de escalado automático para utilizar ponderación de instancias
- Utilización de plantillas de lanzamiento múltiples

Información general de la configuración de creación de un grupo de instancias mixtas

En este tema se proporciona información general y prácticas recomendadas para crear un grupo de escalado automático de instancias mixtas.

Contenido

- Descripción general de
- Flexibilidad del tipo de instancias
- Flexibilidad de zona de disponibilidad
- precio máximo de spot
- Reequilibrio de la capacidad proactivo
- Comportamiento del escalado
- Disponibilidad regional de los tipos de instancias
- Recursos relacionados
- Limitaciones

Descripción general de

Para crear un grupo de instancias mixtas, tiene dos opciones:

- Selección del tipo de instancia basada en atributos: defina sus requisitos de computación para elegir los tipos de instancia automáticamente en función de los atributos específicos de las instancias.
- Selección manual del tipo de instancia: elija manualmente los tipos de instancia que se adapten a su carga de trabajo.

Manual selection

En los siguientes pasos, se describe cómo crear un grupo de instancias mixtas mediante la elección manual de tipos de instancias:

1. Elija una plantilla de lanzamiento que tenga los parámetros para lanzar una instancia de EC2. Los parámetros en las plantillas de lanzamiento son opcionales, pero Amazon EC2 Auto Scaling no puede lanzar una instancia si falta el ID de amilong (AMI) en la plantilla de lanzamiento.
2. Elija la opción para anular la plantilla de lanzamiento.

3. Elija manualmente los tipos de instancias que se adapten a su carga de trabajo.
4. Especificar los porcentajes de las instancias bajo demanda y las instancias de spot que se van a lanzar.
5. Seleccione las estrategias de asignación que determinan cómo Amazon EC2 Auto Scaling satisface las capacidades a pedido y de spot de los posibles tipos de instancias.
6. Elija las zonas de disponibilidad y las subredes de la VPC en las que lanzar las instancias.
7. Especifique el tamaño inicial del grupo (la capacidad deseada) y el tamaño mínimo y máximo del grupo.

Las anulaciones son necesarias para anular el tipo de instancia declarado en la plantilla de lanzamiento y utilizar varios tipos de instancias que estén integrados en la propia definición de recursos del grupo de escalado automático. Para obtener más información acerca de los tipos de instancias que hay disponibles, consulte Tipos de instancias en la Guía del usuario de Amazon EC2.

También puede configurar los siguientes parámetros opcionales para cada tipo de instancia:

- `LaunchTemplateSpecification`: puede asignar una plantilla de inicialización diferente a un tipo de instancia según sea necesario. Esta opción no está disponible desde la consola en este momento. Para obtener más información, consulte Utilización de plantillas de lanzamiento múltiples.
- `WeightedCapacity`: usted decide cuánto representa la instancia para la capacidad deseada en relación con el resto de las instancias de su grupo. Si especifica un valor de `WeightedCapacity` para un tipo de instancias, debe especificar un valor de `WeightedCapacity` para todos ellos. De forma predeterminada, cada instancia cuenta como una instancia para la capacidad deseada. Para obtener más información, consulte Configuración de un grupo de escalado automático para utilizar ponderación de instancias.

Attribute-based selection

Para permitir que Amazon EC2 Auto Scaling elija los tipos de instancia automáticamente en función de sus atributos de instancia específicos, utilice los siguientes pasos para crear un grupo de instancias mixtas especificando sus requisitos de procesamiento:

1. Elija una plantilla de lanzamiento que tenga los parámetros para lanzar una instancia de EC2. Los parámetros en las plantillas de lanzamiento son opcionales, pero Amazon EC2

Auto Scaling no puede lanzar una instancia si falta el ID de amilong (AMI) en la plantilla de lanzamiento.

2. Elija la opción para anular la plantilla de lanzamiento.
3. Especifique los atributos de la instancia que coincidan con sus requisitos de procesamiento, como los requisitos de memoria y v. CPUs
4. Especificar los porcentajes de las instancias bajo demanda y las instancias de spot que se van a lanzar.
5. Seleccione las estrategias de asignación que determinan cómo Amazon EC2 Auto Scaling satisface las capacidades a pedido y de spot de los posibles tipos de instancias.
6. Elija las zonas de disponibilidad y las subredes de la VPC en las que lanzar las instancias.
7. Especifique el tamaño inicial del grupo (la capacidad deseada) y el tamaño mínimo y máximo del grupo.

Las anulaciones son necesarias para anular el tipo de instancia declarado en la plantilla de lanzamiento y utilizar un conjunto de atributos de instancia que describan sus requisitos de procesamiento. Para conocer los atributos compatibles, consulte InstanceRequirementsla referencia de la API Amazon EC2 Auto Scaling. También puede utilizar una plantilla de lanzamiento que ya tenga la definición de atributos de instancia.

También puede configurar el parámetro `LaunchTemplateSpecification` dentro de la estructura de anulaciones para asignar una plantilla de inicialización diferente a un conjunto de requisitos de instancia, según sea necesario. Esta opción no está disponible desde la consola en este momento. Para obtener más información, consulte la LaunchTemplateOverridesreferencia de la API Amazon EC2 Auto Scaling.

De forma predeterminada, establece la cantidad de instancias como la capacidad deseada de su grupo de escalado automático.

Como alternativa, puede establecer el valor de la capacidad deseada en el número de v CPUs o la cantidad de memoria. Para ello, utilice la propiedad `DesiredCapacityType` de la operación de la API `CreateAutoScalingGroup` o el campo desplegable del Tipo de capacidad deseado en la Consola de administración de AWS. Esta es una alternativa útil a las ponderaciones de las instancias.

Flexibilidad del tipo de instancias

Para mejorar la disponibilidad, implemente la aplicación en varios tipos de instancias. Se recomienda utilizar varios tipos de instancias para satisfacer los requisitos de capacidad. De esta forma, Amazon EC2 Auto Scaling puede lanzar otro tipo de instancia si no hay suficiente capacidad en las zonas de disponibilidad elegidas.

Si no hay suficiente capacidad de instancia con las instancias de spot, Amazon EC2 Auto Scaling sigue intentando lanzarlas desde otros grupos de instancias spot. (Los grupos que usa están determinados por la elección de los tipos de instancias y la estrategia de asignación). Amazon EC2 Auto Scaling le permite aprovechar el ahorro de costos de las instancias de spot lanzándolas en lugar de las instancias bajo demanda.

Recomendamos ser flexible con al menos 10 tipos de instancias para cada carga de trabajo. Al elegir los tipos de instancias, no se limite a los nuevos tipos de instancias más populares. Elegir tipos de instancias de generación anterior tiende a provocar menos interrupciones de spot porque tienen menos demanda de los clientes bajo demanda.

Flexibilidad de zona de disponibilidad

Recomendamos ampliamente que extienda el grupo de escalado automático en varias zonas de disponibilidad. Con varias zonas de disponibilidad, puede diseñar aplicaciones que realizan una conmutación por error automática entre zonas para aumentar la resiliencia.

Como beneficio adicional, puede acceder a un grupo de capacidad de Amazon EC2 más amplio en comparación con los grupos de una sola zona de disponibilidad. Dado que la capacidad fluctúa de manera independiente para cada tipo de instancias en una zona de disponibilidad, a menudo puede obtener más capacidad informática con flexibilidad en el tipo de instancias y la zona de disponibilidad.

Para obtener más información acerca del uso de varias zonas de disponibilidad, consulte Ejemplo: distribuir instancias entre zonas de disponibilidad.

precio máximo de spot

Al crear el grupo de Auto Scaling con el AWS CLI o un SDK, puede especificar el `SpotMaxPrice` parámetro. El parámetro `SpotMaxPrice` determina el precio máximo que está dispuesto a pagar por una hora de instancia de spot.

Al especificar el parámetro `WeightedCapacity` en las anulaciones (o `"DesiredCapacityType": "vcpu"` o `"DesiredCapacityType": "memory-mib"` a nivel de grupo), el precio máximo representa el precio unitario máximo, no el precio máximo de una instancia completa.

Le recomendamos enfáticamente que no especifique un precio máximo. Su aplicación podría no ejecutarse si no recibe sus instancias de spot, como cuando el precio máximo es demasiado bajo. Si no especifica un precio máximo, el predeterminado es el precio bajo demanda. Solo pagará el precio de spot de las instancias de spot que lance. Seguirá recibiendo grandes descuentos por parte de las instancias de spot. Estos descuentos son posibles debido a los precios de spot estables disponibles mediante el modelo de precios de las instancias de spot. Para obtener más información, consulte Precios y ahorro en la Guía del usuario de Amazon EC2.

Reequilibrio de la capacidad proactivo

Si su caso de uso lo permite, le recomendamos el reequilibrio de la capacidad. El reequilibrio de capacidad permite mantener la disponibilidad de la carga de trabajo al sustituir de forma proactiva las instancias de spot que corren el riesgo de sufrir interrupciones.

Cuando se habilita el reequilibrio de la capacidad, Amazon EC2 Auto Scaling intenta reemplazar de forma proactiva las instancias de spot que han recibido una recomendación de reequilibrio. Esto da la oportunidad de reequilibrar la carga de trabajo con nuevas instancias de spot que no tengan un riesgo elevado de interrupción.

Para obtener más información, consulte Reequilibrio de la capacidad en Auto Scaling para reemplazar las instancias de spot en riesgo.

Comportamiento del escalado

Al crear un grupo de instancias mixtas, este utiliza instancias bajo demanda de forma predeterminada. Para utilizar instancias de spot, debe modificar el porcentaje del grupo que lanzar como instancias bajo demanda. Puede especificar cualquier número del 0 al 100 para el porcentaje bajo demanda.

De forma opcional, también puede designar un número base de instancias bajo demanda para comenzar. Si lo hace, Amazon EC2 Auto Scaling no lanza las instancias de spot hasta que se lance la capacidad base de instancias bajo demanda cuando el grupo escale horizontalmente. Lo que esté más allá de la capacidad base utiliza porcentajes bajo demanda para determinar cuántas instancias bajo demanda y de spot se deben lanzar.

Amazon EC2 Auto Scaling convierte el porcentaje en el número equivalente de instancias. Si el resultado crea un número fraccionario, redondea al siguiente número entero a favor de las instancias bajo demanda.

La tabla siguiente muestra el comportamiento del grupo de escalado automático a medida que reduce y aumenta su tamaño.

Ejemplo: Comportamiento del escalado

Opciones de compra	Tamaño de grupo y número de instancias de ejecución en las opciones de compra			
	10	20	30	40
Ejemplo 1: base de 10, 50/50 % bajo demanda/spot				
Instancias bajo demanda (importe base)	10	10	10	10
instancias bajo demanda	0	5	10	15
Spot Instances	0	5	10	15
Ejemplo 2: base de 0, 0/100 % bajo demanda/spot				
Instancias bajo demanda (importe base)	0	0	0	0
instancias bajo demanda	0	0	0	0

Opciones de compra	Tamaño de grupo y número de instancias de ejecución en las opciones de compra			
Spot Instances	10	20	30	40
Ejemplo 3: base de 0, 60/40 % bajo demanda/spot				
Instancias bajo demanda (importe base)	0	0	0	0
instancias bajo demanda	6	12	18	24
Spot Instances	4	8	12	16
Ejemplo 4: base de 0, 100/0 % bajo demanda/spot				
Instancias bajo demanda (importe base)	0	0	0	0
instancias bajo demanda	10	20	30	40
Spot Instances	0	0	0	0
Ejemplo 5: base de 12, 0/100 % bajo demanda/spot				

Opciones de compra	Tamaño de grupo y número de instancias de ejecución en las opciones de compra			
Instancias bajo demanda (importe base)	10	12	12	12
instancias bajo demanda	0	0	0	0
Spot Instances	0	8	18	28

Cuando el tamaño del grupo aumenta, Amazon EC2 Auto Scaling intenta equilibrar su capacidad de manera uniforme en las zonas de disponibilidad especificadas. A continuación, lanza tipos de instancia de acuerdo con la estrategia de asignación especificada.

Cuando el tamaño del grupo disminuye, Amazon EC2 Auto Scaling identifica primero cuál de los dos tipos (spot o bajo demanda) debe ser terminado. A continuación, intenta terminar las instancias de forma equilibrada en todas las zonas de disponibilidad especificadas. También favorece la finalización de las instancias de una forma que se ajuste más a sus estrategias de asignación. Para obtener más información sobre las políticas de terminación, consulte Configuración de políticas de terminación de Amazon EC2 Auto Scaling.

Disponibilidad regional de los tipos de instancias

La disponibilidad de los tipos de instancias de EC2 varía en función del usuario. Región de AWS Por ejemplo, es posible que los tipos de instancias de última generación aún no estén disponibles en una región determinada. Debido a las variaciones en la disponibilidad de las instancias de una región a otra, es posible que tenga problemas a la hora de realizar solicitudes programáticas si en su región no están disponibles varios tipos de instancias en sus anulaciones. Si utiliza varios tipos de instancias que no estén disponibles en su región, la solicitud podría fallar por completo. Para resolver el problema, vuelva a intentar la solicitud con distintos tipos de instancias y asegúrese de que cada tipo de instancia esté disponible en la región. Para buscar los tipos de instancias que se ofrecen por ubicación, utilice el describe-instance-type-offeringscomando. Para obtener más información, consulte Finding an Amazon EC2 instance type en la Guía del usuario de Amazon EC2.

Recursos relacionados

Para obtener más prácticas recomendadas para instancias de spot, consulte Prácticas recomendadas para instancias de spot de EC2 en la Guía del usuario de Amazon EC2.

Limitaciones

Después de agregar anulaciones a un grupo de escalado automático mediante una política de instancias mixtas, puede actualizar las anulaciones con la llamada a la API `UpdateAutoScalingGroup`, pero no eliminarlas. Para eliminar por completo las anulaciones, primero debe cambiar el grupo de escalado automático para utilizar una plantilla o una configuración de inicialización en lugar de una política de instancias mixtas. A continuación, puede volver a añadir una política de instancias mixtas sin anularla.

Estrategias de asignación de varios tipos de instancias

Cuando utiliza varios tipos de instancias, administra cómo Amazon EC2 Auto Scaling satisface sus capacidades bajo demanda y de spot de los posibles tipos de instancias. Para ello, debe especificar estrategias de asignación.

Para revisar las prácticas recomendadas para un grupo de instancias mixtas, consulte Información general de la configuración de creación de un grupo de instancias mixtas.

Contenido

- Spot Instances
- instancias bajo demanda
- Cómo funcionan las estrategias de asignación con las ponderaciones

Spot Instances

Amazon EC2 Auto Scaling ofrece las siguientes estrategias de asignación para instancias de spot:

`price-capacity-optimized` (recomendado)

> La estrategia de asignación optimizada por precio y capacidad analiza tanto el precio como la capacidad para seleccionar los grupos de instancias de spot que tienen menos probabilidades de interrupción y el precio más bajo posible.

Le recomendamos esta estrategia cuando empiece. Para obtener más información, consulte Introducción a la estrategia de price-capacity-optimized asignación para las instancias puntuales de EC2 en el AWS blog.

`capacity-optimized`

Amazon EC2 Auto Scaling solicita las instancias de spot desde el grupo con capacidad óptima para el número de instancias que va a lanzar.

Con las instancias de spot, los precios cambian lentamente en función de tendencias a largo plazo registradas en la oferta y la demanda. Sin embargo, la capacidad fluctúa en tiempo real. La estrategia `capacity-optimized` inicia instancias de spot de forma automática en los grupos con mayor disponibilidad, analizando los datos de capacidad en tiempo real y prediciendo cuáles son los que tienen una mayor disponibilidad. Esto ayuda a minimizar las posibles interrupciones de trabajo que pueden tener un costo de interrupción superior asociado al reinicio del trabajo y la creación de puntos de control. Para brindar una mayor probabilidad de lanzar primero a ciertos tipos de instancia, utilice `capacity-optimized-prioritized`.

`capacity-optimized-prioritized`

Se establece el orden de los tipos de instancia para las anulaciones de plantillas de lanzamiento de mayor a menor prioridad (de la primera a la última de la lista). Amazon EC2 Auto Scaling respeta las prioridades de tipo de instancia haciendo todo lo posible, pero optimiza primero la capacidad. Esta es una buena opción para cargas de trabajo en las que se debe minimizar la posibilidad de interrupción, pero también importa la preferencia por ciertos tipos de instancia. Si la estrategia de asignación bajo demanda se establece en `prioritized`, se aplica la misma prioridad cuando se completa la capacidad bajo demanda.

`lowest-price` (no se recomienda)

> **Warning**
>
> No recomendamos la estrategia `lowest-price` porque presenta el mayor riesgo de interrupción para las instancias de spot.

Amazon EC2 Auto Scaling solicita sus instancias de spot mediante los grupos con el precio más bajo dentro de una zona de disponibilidad, en el número N de grupos de spot que especifique para la configuración de Grupos más baratos. Por ejemplo, si especifica cuatro tipos de instancia y cuatro zonas de disponibilidad, el grupo de escalado automático puede acceder a un máximo

de 16 grupos de spot. (Cuatro en cada zona de disponibilidad). Si especifica dos grupos de spot (N = 2) para la estrategia de asignación, el grupo de escalado automático puede recurrir a los dos grupos más baratos por zona de disponibilidad para cumplir con su capacidad de spot.

La estrategia `lowest-price` solo está disponible cuando se utiliza la AWS CLI.

Amazon EC2 Auto Scaling hace el esfuerzo de obtener instancias de spot del número N de grupos que especifique. Sin embargo, si un grupo se queda sin capacidad de spot antes de cubrir su capacidad deseada, Amazon EC2 Auto Scaling continúa cumpliendo con su solicitud extrayendo capacidad del siguiente grupo más barato. Para que se logre la capacidad deseada, es posible que reciba instancias de spot de una cantidad de grupos mayor al número N de grupos que especificó. Del mismo modo, si la mayoría de los grupos no tienen capacidad de spot, es posible que reciba su capacidad deseada total de menos grupos que el número N de grupos que especificó.

> **Note**
>
> Si configura la instancia de spot para lanzarla con la característica SEV-SNP de AMD activada, se le cobrará una tarifa de uso por hora adicional que equivale al 10 % de la tarifa horaria bajo demanda del tipo de instancia seleccionado. Si la estrategia de asignación utiliza el precio como variable, Amazon EC2 Auto Scaling no incluye esta tarifa adicional; solo se utiliza el precio de spot.

instancias bajo demanda

Amazon EC2 Auto Scaling ofrece las siguientes estrategias de asignación que se pueden utilizar para instancias bajo demanda:

`lowest-price`

Amazon EC2 Auto Scaling implementa automáticamente el tipo de instancia más barato en cada zona de disponibilidad según el precio bajo demanda actual.

Para garantizar que se logre la capacidad deseada, es posible que reciba instancias bajo demanda de más de un tipo de instancia en cada zona de disponibilidad. Esto depende de la cantidad de capacidad que solicite.

`prioritized`

Amazon EC2 Auto Scaling utiliza el orden de los tipos de instancia en la lista de anulaciones de la plantilla de lanzamiento para determinar qué tipo de instancia se utilizará en primer lugar cuando se cumpla con la capacidad bajo demanda. Por ejemplo, supongamos que se especifican tres anulaciones de plantilla de lanzamiento en el siguiente orden: `c5.large`, `c4.large` y `c3.large`. Cuando se lanzan las instancias bajo demanda, el grupo de escalado automático satisface la capacidad bajo demanda en el siguiente orden: `c5.large`, `c4.large` y luego `c3.large`.

Tenga en cuenta lo siguiente cuando administre el orden de prioridad de las instancias a petición:

- Puede pagar el uso por adelantado para conseguir importantes descuentos en las instancias bajo demanda a través de Savings Plans o las instancias reservadas. Para obtener más información, consulte la página Precios de Amazon EC2.
- Con las instancias reservadas, se aplicará un descuento sobre los precios normales de las instancias en diferido si Amazon EC2 Auto Scaling lanza tipos de instancias coincidentes. Esto significa que, si tiene instancias reservadas de `c4.large` sin utilizar, puede establecer la prioridad del tipo de instancia para asignar la prioridad más alta de sus instancias reservadas a un tipo de instancia `c4.large`. Cuando se lanza una instancia `c4.large`, obtendrá el precio de instancia reservada.
- Con Savings Plans, el descuento sobre los precios normales de las instancias bajo demanda se aplica cuando se utilizan Savings Plans para instancias de Amazon EC2 o Savings Plans para computación. Con Savings Plans, tiene más flexibilidad a la hora de priorizar los tipos de instancia. Siempre que utilice tipos de instancia que estén incluidos en su Savings Plans, puede asignarles cualquier orden de prioridad. Ocasionalmente, puede, además, cambiar todo el orden de los tipos de instancia y seguir beneficiándose de la tarifa de Savings Plans con descuento. Para obtener más información sobre Savings Plans, consulte la Guía del usuario de Savings Plans.

Cómo funcionan las estrategias de asignación con las ponderaciones

Cuando especifique el parámetro `WeightedCapacity` en las anulaciones (o `"DesiredCapacityType": "vcpu"` o `"DesiredCapacityType": "memory-mib"` a nivel de grupo), las estrategias de asignación funcionan exactamente igual que con otros grupos de escalado automático.

Suponga que tiene un grupo de Auto Scaling con varios tipos de instancias que tienen cantidades variables de vCPUs. Lo utiliza `lowest-price` para sus estrategias de asignación puntual y bajo demanda. Si elige asignar las ponderaciones en función del recuento de vCPU de cada tipo de instancia, Amazon EC2 Auto Scaling lanza cualquier tipo de instancia que tenga el precio más bajo según los valores de ponderación asignados (por ejemplo, por vCPU) al momento de ejecutarse. Si se trata de una instancia de spot, significa el precio de spot más bajo por vCPU. Si se trata de una instancia bajo demanda, significa el precio bajo demanda más bajo por vCPU.

Para obtener más información, consulte Configuración de un grupo de escalado automático para utilizar ponderación de instancias.

Creación de grupos de instancias mixtas mediante la selección del tipo de instancia basada en atributos

En lugar de usar la elección manual de los tipos de instancia para un grupo de instancias mixtas, puede especificar un conjunto de atributos de instancia que describan los requisitos de computación. A medida que Amazon EC2 Auto Scaling lanza instancias, todos los tipos de instancia que utiliza el grupo de escalado automático deben coincidir con los atributos de instancia requeridos. Esto se conoce como selección del tipo de instancia basada en atributos.

Este enfoque es ideal para las cargas de trabajo y los marcos que tienen flexibilidad en cuanto a qué tipos de instancia utilizan, tales como contenedores, macrodatos y CI/CD.

A continuación, se describen los beneficios de la selección del tipo de instancia basada en atributos:

- Flexibilidad óptima para instancias de spot: Amazon EC2 Auto Scaling puede seleccionar entre una amplia gama de tipos de instancia para inicializar instancias de spot. Esto se realiza conforme a la práctica recomendada de spot que consiste en ser flexible en cuanto a los tipos de instancia, lo que brinda al servicio de spot de Amazon EC2 más posibilidades de encontrar y asignar la cantidad necesaria de capacidad de computación.
- Uso fácil de los tipos de instancias correctos: con tantos tipos de instancias disponibles, encontrar los tipos de instancias adecuados para su carga de trabajo puede necesitar mucho tiempo. Cuando especifica atributos de instancia, los tipos de instancia tendrán automáticamente los atributos necesarios para la carga de trabajo.
- Uso automático de nuevos tipos de instancias: los grupos de escalado automático pueden utilizar tipos de instancias de última generación a medida que se inicializan. Los tipos de instancias de última generación se utilizan de forma automática cuando coinciden con sus requisitos y se ajustan a las estrategias de asignación que elija para su grupo de escalado automático.

Temas

- Cómo funciona la selección de tipo de instancia basada en atributos
- Protección de precios
- Protección del rendimiento
- Requisitos previos
- Creación de un grupo de instancias mixtas mediante la selección del tipo de instancia basada en atributos (consola)
- Creación de un grupo de instancias mixtas mediante la selección del tipo de instancia basada en atributos (AWS CLI)
- Configuración de ejemplo
- Obtención de una vista previa de los tipos de instancia
- Recursos relacionados

Cómo funciona la selección de tipo de instancia basada en atributos

Con la selección de tipo de instancia basada en atributos, en lugar de proporcionar una lista de tipos de instancias específicos, proporciona una lista de atributos que requieren sus instancias, como:

- Recuento de vCPU: el número mínimo y máximo de v CPUs por instancia.
- Memoria: el mínimo y el máximo GiBs de memoria por instancia.
- Almacenamiento local: si se usarán volúmenes de almacén de instancias o EBS para el almacenamiento local.
- Rendimiento ampliable: si se usará la familia de instancias T, incluidos los tipos T4g, T3a, T3 y T2.

Hay muchas opciones disponibles para definir los requisitos de las instancias. Para obtener una descripción de cada opción y los valores predeterminados, consulte la referencia InstanceRequirementsde la API Amazon EC2 Auto Scaling.

Cuando su grupo de escalado automático necesite inicializar una instancia, buscará los tipos de instancia que coincidan con los atributos especificados y que estén disponibles en esa zona de disponibilidad. Las estrategias de asignación luego determinan los tipos de instancia que se van a inicializar. De forma predeterminada, la selección del tipo de instancia basada en atributos tiene habilitada una característica de protección de precios para evitar que su grupo de escalado automático inicialice tipos de instancias que superen los umbrales de su presupuesto.

De forma predeterminada, utiliza la cantidad de instancias como la unidad de medida cuando establece la capacidad deseada de su grupo de escalado automático, lo que significa que cada instancia cuenta como una unidad.

Como alternativa, puede establecer el valor de la capacidad deseada en el número de v CPUs o la cantidad de memoria. Para ello, utilice el campo desplegable del tipo de capacidad deseado en la operación Consola de administración de AWS o de la `UpdateAutoScalingGroup` API `CreateAutoScalingGroup` o. `DesiredCapacityType` A continuación, Amazon EC2 Auto Scaling inicializa la cantidad de instancias necesaria para cumplir con la capacidad de vCPU o memoria deseada. Por ejemplo, si usa v CPUs como el tipo de capacidad deseado y usa instancias con 2 v CPUs cada una, una capacidad deseada de 10 v CPUs lanzaría 5 instancias. Esta es una alternativa útil a las ponderaciones de las instancias.

Protección de precios

Con la protección de precios, puede especificar el precio máximo que está dispuesto a pagar por las instancias de EC2 inicializadas por su grupo de escalado automático. La protección de precios es una característica que impide que su grupo de escalado automático utilice tipos de instancias que consideraría demasiado caros, incluso si se ajustan a los atributos especificados.

La protección de precios está habilitada de forma predeterminada y tiene umbrales de precios diferentes para las instancias bajo demanda y las instancias de spot. Cuando Amazon EC2 Auto Scaling necesita inicializar nuevas instancias, no se inicializa ningún tipo de instancia cuyo precio supere el umbral correspondiente.

Temas

- Protección de precios bajo demanda
- Protección de precios de spot
- Personalización de la protección de precios

Protección de precios bajo demanda

En el caso de las instancias bajo demanda, usted define el precio máximo bajo demanda que está dispuesto a pagar como un porcentaje superior al precio bajo demanda identificado. El precio bajo demanda identificado es el precio del tipo de instancia C, M o R de la generación actual más barato con sus atributos especificados.

Si no se define explícitamente un valor de protección de precios bajo demanda, se utilizará un precio máximo bajo demanda predeterminado un 20 por ciento superior al precio bajo demanda identificado.

Protección de precios de spot

De manera predeterminada, Amazon EC2 Auto Scaling aplicará automáticamente una protección óptima del precio de las instancias de spot para seleccionar de manera coherente entre una amplia gama de tipos de instancia. También puede configurar manualmente la protección de precios. Sin embargo, dejar que Amazon EC2 Auto Scaling lo haga puede aumentar la probabilidad de que se agote la capacidad de spot.

Puede especificar manualmente la protección de precios con una de las opciones siguientes. Si configura manualmente la protección de precios, le recomendamos utilizar la primera opción.

- Un porcentaje de un precio identificado bajo demanda: el precio identificado bajo demanda es el precio del tipo de instancia C, M o R de la generación actual con el precio más bajo, con los atributos especificados.
- Un porcentaje superior al precio de spot identificado: el precio de spot identificado es el precio del tipo de instancia C, M o R de la generación actual con el precio más bajo, con sus atributos especificados. No le recomendamos utilizar esta opción porque los precios de spot pueden fluctuar y, por lo tanto, su umbral de protección de precios también.

Personalización de la protección de precios

Puede personalizar los umbrales de protección de precios en la consola Amazon EC2 Auto Scaling o utilizando AWS CLI o. SDKs

- En la consola, utilice las configuraciones de protección de precios bajo demanda y protección de precios de spot en los atributos de instancias adicionales.
- En la InstanceRequirementsestructura, para especificar el umbral de protección del precio de las instancias bajo demanda, utilice la `OnDemandMaxPricePercentageOverLowestPrice` propiedad. Para especificar el umbral de protección del precio de las instancias de spot, utilice la propiedad `MaxSpotPriceAsPercentageOfOptimalOnDemandPrice` o `SpotMaxPricePercentageOverLowestPrice`.

Si estableces el tipo de capacidad deseado (`DesiredCapacityType`) en v CPUs o GiB de memoria, la protección de precios se aplica en función del precio por vCPU o memoria, en lugar del precio por instancia.

También puede desactivar la protección de precios. Para indicar que no hay umbral de protección de precios, especifique un valor de porcentaje alto, como `999999`.

> **Note**
>
> Si ningún tipo de instancia C, M o R actual de la generación actual coincide con los atributos especificados, la protección de precios sigue vigente. Si no se encuentra ninguna coincidencia, el precio identificado corresponde a los tipos de instancias de la generación actual con el precio más bajo o, en su defecto, a los tipos de instancias de la generación anterior con el precio más bajo que coincidan con sus atributos.

Protección del rendimiento

La protección de rendimiento es una característica que asegura que su grupo de escalado automático utilice tipos de instancias similares o superiores a un rendimiento base especificado. Para utilizar la protección del rendimiento, debe especificar una familia de instancias como referencia base. Las capacidades de la familia de instancias especificada definen el nivel mínimo aceptable de rendimiento. Cuando Auto Scaling selecciona los tipos de instancias, tiene en cuenta los atributos especificados y el rendimiento base especificado. Los tipos de instancias que no cumplen con la referencia de rendimiento se excluyen automáticamente de la selección, incluso si coinciden con los demás atributos especificados. De este modo se garantiza que todos los tipos de instancias seleccionados ofrezcan un rendimiento similar o superior a la referencia establecida por la familia de instancias especificada. Auto Scaling utiliza esta referencia para orientar la selección del tipo de instancia, pero no se garantiza que los tipos de instancia seleccionados superen siempre la referencia para cada aplicación.

Actualmente, esta característica solo admite el rendimiento de la CPU como factor de rendimiento de referencia. El rendimiento de la CPU de la familia de instancias especificada sirve como referencia de rendimiento, lo que garantiza que los tipos de instancia seleccionados sean similares a esta referencia o la superen. Las familias de instancias con los mismos procesadores de CPU conducen a los mismos resultados de filtrado, aunque su rendimiento de red o de disco difiera. Por ejemplo, si se especifica `c6in` o `c6i` como referencia base, se obtendrían resultados de filtrado basados en el rendimiento idénticos, ya que ambas familias de instancias utilizan el mismo procesador de CPU.

Familias de instancias no admitidas

No se admiten las siguientes familias de instancias para la protección del rendimiento:

- c1
- g3 | g3s
- hpc7g
- m1 | m2
- mac1 | mac2 | mac2-m1ultra | mac2-m2 | mac2-m2pro
- p3dn | p4d | p5
- t1
- u-12tb1 | u-18tb1 | u-24tb1 | u-3tb1 | u-6tb1 | u-9tb1 | u7i-12tb | u7in-16tb | u7in-24tb | u7in-32tb

Si para habilitar la protección del rendimiento, especifica una familia de instancias admitida, los tipos de instancias devueltos excluirán las familias de instancias no admitidas que se indican más arriba.

Ejemplo: establecimiento de una referencia de rendimiento base de la CPU

En el siguiente ejemplo, el requisito de instancias consiste en realizar el lanzamiento con tipos de instancias que tengan núcleos de CPU con el mismo rendimiento que la familia de instancias de c6i. Esto filtrará los tipos de instancias con procesadores de CPU de menor rendimiento, incluso si cumplen con los demás requisitos de instancia especificados, como el número de v. CPUs Por ejemplo, si los atributos de instancia que especificaste incluyen 4 v CPUs y 16 GB de memoria, se c6i excluirá de la selección un tipo de instancia con estos atributos pero con un rendimiento de CPU inferior al de estos.

```
"BaselinePerformanceFactors": {
        "Cpu": {
            "References": [
                {
                    "InstanceFamily": "c6i"
                }
            ]
        }
```

Consideraciones

Cuando utilice la protección del rendimiento, tenga en cuenta lo siguiente:

- Puede especificar tipos de instancias o atributos de instancia, pero no ambos al mismo tiempo.
- Puede especificar un máximo de cuatro estructuras `InstanceRequirements` en una configuración de solicitud.

Requisitos previos

- Cree una plantilla de lanzamiento. Para obtener más información, consulte Creación de una plantilla de lanzamiento para un grupo de Auto Scaling.
- Verifique que la plantilla de lanzamiento no solicite ya instancias de spot.

Creación de un grupo de instancias mixtas mediante la selección del tipo de instancia basada en atributos (consola)

Utilice el siguiente procedimiento para crear un grupo de instancias mixtas mediante la selección del tipo de instancia basada en atributos. Para ayudarlo a realizar los pasos de forma eficiente, se omiten algunas secciones opcionales.

Para la mayoría de las cargas de trabajo de uso general, basta con especificar la cantidad de v CPUs y la memoria que necesitas. Para los casos de uso avanzados, puede especificar atributos como el tipo de almacenamiento, las interfaces de red, el fabricante de la CPU y el tipo de acelerador.

Para revisar las prácticas recomendadas para un grupo de instancias mixtas, consulte Información general de la configuración de creación de un grupo de instancias mixtas.

Creación de un grupo de instancias mixtas

1. Abra la consola Amazon EC2 en https://console.aws.amazon.com/ec2/y seleccione Auto Scaling Groups en el panel de navegación.
2. En la barra de navegación de la parte superior de la pantalla, elija la misma Región de AWS que utilizó cuando creó la plantilla de lanzamiento.
3. Elija Create an Auto Scaling group (Crear un grupo de escalado automático).
4. En la página Choose launch template or configuration (Elegir una plantilla de lanzamiento o configuración), ingrese un nombre para el grupo de escalado automático.
5. Para elegir la plantilla de lanzamiento, haga lo siguiente:

    a. En launch template (Plantilla de lanzamiento), elija una plantilla de lanzamiento existente.

b. Para Launch template version (Versión de plantilla de lanzamiento), decida si el grupo de escalado automático utiliza el valor predeterminado, la última versión o una versión específica de la plantilla de lanzamiento para escalado horizontal.

c. Compruebe que la plantilla de lanzamiento admita todas las opciones que tiene previsto utilizar y, a continuación, elija Next (Siguiente).

6. En la página Elegir las opciones de lanzamiento de instancias, haga lo siguiente:

a. En Instance type requirements (Requisitos de los tipos de instancia), elija Override launch template (Anular la plantilla de lanzamiento).

> **Note**
>
> Si eligió una plantilla de lanzamiento que ya contiene un conjunto de atributos de instancia, como las vCPU y la memoria, se muestran los atributos de la instancia. Estos atributos se añaden a las propiedades del grupo de escalado automático, donde puede actualizarlos desde la consola de escalado automático de Amazon EC2 Auto Scaling, en cualquier momento.

b. En Especificar los atributos de la instancia, comience por introducir su v CPUs y sus requisitos de memoria.

- Para v CPUs, introduce el número mínimo y máximo deseado de vCPUs. Para no especificar ningún límite, seleccione Sin mínimo, Sin máximo o ambos.
- En Memory (GiB) (Memoria [GiB]), ingrese la cantidad mínima y máxima de memoria deseada. Para no especificar ningún límite, seleccione No minimum (Sin mínimo), No maximum (Sin máximo) o ambos.

c. (Opcional) En Additional instance attributes (Atributos de instancia adicionales), puede especificar opcionalmente uno o varios atributos para expresar sus requisitos de computación con más detalle. Cada atributo adicional agrega más restricciones a la solicitud.

d. Expanda Vista previa de tipos de instancia coincidentes para ver los tipos de instancias que tienen los atributos especificados.

e. En Opciones de compra de instancias, para la Distribución de instancias, especifique los porcentajes del grupo que se lanzará como instancias bajo demanda e instancias de spot, respectivamente. Si la suya es una aplicación sin estado, tolerante a errores y capaz

de gestionar la interrupción de una instancia, puede especificar un mayor porcentaje de instancias de spot.

f. (Opcional) Cuando especifique un porcentaje para las instancias de spot, seleccione Incluir capacidad base bajo demanda y luego especifique la cantidad mínima de la capacidad inicial del grupo de escalado automático que deben satisfacer las instancias bajo demanda. Lo que esté más allá de la capacidad base utiliza la configuración de Instances distribution (Distribución de las instancias) para determinar cuántas instancias bajo demanda y de spot deben lanzarse.

g. En Allocation strategies (Estrategias de asignación), se selecciona de forma automática la opción Lowest price (Precio más bajo) para On-Demand allocation strategy (Estrategia de asignación bajo demanda) y no es posible cambiarla.

h. En Spot allocation strategy (Estrategia de asignación de spot), elija una estrategia de asignación. Price capacity optimized (Capacidad de precios optimizada) se selecciona de forma predeterminada.

i. En Reequilibrio de la capacidad, elija si desea habilitar o desactivar el reequilibrio de la capacidad. Use el reequilibrio de la capacidad para responder automáticamente cuando sus instancias de spot se aproximen a su finalización por una interrupción de spot. Para obtener más información, consulte Reequilibrio de la capacidad en Auto Scaling para reemplazar las instancias de spot en riesgo.

j. En Network (Red), para la opción VPC, elija una VPC. El grupo de escalado automático debe crearse en la misma VPC que el grupo de seguridad especificado en la plantilla de lanzamiento.

k. En Availability Zones and subnets (Zonas de disponibilidad y subredes), elija una o más subredes de la VPC especificada. Utilice subredes en varias zonas de disponibilidad para lograr una alta disponibilidad. Para obtener más información, consulte Consideraciones a la hora de elegir subredes de VPC.

l. Elija Siguiente, Siguiente.

7. Para el paso Configure group size and scaling policies (Configurar el tamaño del grupo y las políticas de escalado), haga lo siguiente:

a. Para medir la capacidad deseada en unidades distintas de las instancias, elija la opción adecuada para Tamaño del grupo, Tipo de capacidad deseada. Se admiten unidadesCPUs, v y GiB de memoria. De forma predeterminada, Amazon EC2 Auto Scaling especifica Units (Unidades), lo que se traduce en número de instancias.

b. Para la Capacidad deseada, establezca el tamaño inicial de su grupo de escalado automático.

c. En la sección Escalado, en Límites de escalado, si el nuevo valor de Capacidad deseada es superior a Capacidad deseada mínima y a Capacidad deseada máxima, Capacidad deseada máxima se incrementará automáticamente al nuevo valor de capacidad deseada. Puede cambiar estos límites según sea necesario. Para obtener más información, consulte Establecimiento de límites de escalado para el grupo de escalado automático.

8. Elija Skip to review (Omitir para revisar).
9. En la página Review (Revisar), elija Create Auto Scaling group (Crear grupo de escalado automático).

Creación de un grupo de instancias mixtas mediante la selección del tipo de instancia basada en atributos (AWS CLI)

Creación de un grupo de instancias mixtas mediante la línea de comandos

Utilice uno de los siguientes comandos:

- create-auto-scaling-group (AWS CLI)
- Nuevo- ASAuto ScalingGroup ()AWS Tools for Windows PowerShell

Configuración de ejemplo

Para crear un grupo de Auto Scaling con una selección de tipo de instancia basada en atributos mediante el AWS CLI, utilice el siguiente create-auto-scaling-groupcomando.

Se especifican los siguientes atributos de instancia:

- `VCpuCount`— Los tipos de instancia deben tener un mínimo de cuatro v CPUs y un máximo de ocho v. CPUs
- `MemoryMiB`: los tipos de instancia deben tener un mínimo de 16 384 MiB de memoria.
- `CpuManufacturers`: los tipos de instancia deben tener una CPU fabricada por Intel.

JSON

```
aws autoscaling create-auto-scaling-group --cli-input-json file://~/config.json
```

A continuación se muestra un ejemplo de un archivo `config.json`.

```
{
    "AutoScalingGroupName": "my-asg",
    "DesiredCapacityType": "units",
    "MixedInstancesPolicy": {
        "LaunchTemplate": {
            "LaunchTemplateSpecification": {
                "LaunchTemplateName": "my-launch-template",
                "Version": "$Default"
            },
            "Overrides": [{
                "InstanceRequirements": {
                    "VCpuCount": {"Min": 4, "Max": 8},
                    "MemoryMiB": {"Min": 16384},
                    "CpuManufacturers": ["intel"]
                }
            }]
        },
        "InstancesDistribution": {
            "OnDemandPercentageAboveBaseCapacity": 50,
            "SpotAllocationStrategy": "price-capacity-optimized"
        }
    },
    "MinSize": 0,
    "MaxSize": 100,
    "DesiredCapacity": 4,
    "DesiredCapacityType": "units",
    "VPCZoneIdentifier": "subnet-5ea0c127,subnet-6194ea3b,subnet-c934b782"
}
```

Para establecer el valor de la capacidad deseada como el número de v CPUs o la cantidad de memoria, especifique `"DesiredCapacityType": "vcpu"` o `"DesiredCapacityType": "memory-mib"` en el archivo. El tipo de capacidad deseada predeterminado es `units`, que establece el valor de la capacidad deseada como la cantidad de instancias.

YAML

Como alternativa, puede usar el siguiente create-auto-scaling-groupcomando para crear el grupo Auto Scaling. Esto hace referencia a un archivo YAML como único parámetro de su grupo de escalado automático.

```
aws autoscaling create-auto-scaling-group --cli-input-yaml file://~/config.yaml
```

A continuación se muestra un ejemplo de un archivo `config.yaml`.

```
---
AutoScalingGroupName: my-asg
DesiredCapacityType: units
MixedInstancesPolicy:
  LaunchTemplate:
    LaunchTemplateSpecification:
      LaunchTemplateName: my-launch-template
      Version: $Default
    Overrides:
    - InstanceRequirements:
        VCpuCount:
          Min: 2
          Max: 4
        MemoryMiB:
          Min: 2048
        CpuManufacturers:
        - intel
  InstancesDistribution:
    OnDemandPercentageAboveBaseCapacity: 50
    SpotAllocationStrategy: price-capacity-optimized
MinSize: 0
MaxSize: 100
DesiredCapacity: 4
DesiredCapacityType: units
VPCZoneIdentifier: subnet-5ea0c127,subnet-6194ea3b,subnet-c934b782
```

Para establecer el valor de la capacidad deseada como el número de v CPUs o la cantidad de memoria, especifique `DesiredCapacityType: vcpu` o `DesiredCapacityType: memory-mib` en el archivo. El tipo de capacidad deseada predeterminado es `units`, que establece el valor de la capacidad deseada como la cantidad de instancias.

Para ver un ejemplo de cómo utilizar varias plantillas de lanzamiento con la selección del tipo de instancia basada en atributos, consulte Utilización de plantillas de lanzamiento múltiples.

Obtención de una vista previa de los tipos de instancia

Puede obtener una vista previa de los tipos de instancia que son compatibles con sus requisitos de computación sin necesidad de lanzarlos y ajustar los requisitos en caso de ser necesario. Cuando

crea el grupo de escalado automático en la consola de Amazon EC2 Auto Scaling, aparece una vista previa de los tipos de instancia en la sección Preview matching instance types (Vista previa de los tipos de instancia coincidentes) en la página Choose instance launch options (Elegir opciones de lanzamiento de las instancias).

Como alternativa, puedes previsualizar los tipos de instancias realizando una llamada a la GetInstanceTypesFromInstanceRequirementsAPI de Amazon EC2 con el AWS CLI o un SDK. Transfiera los parámetros `InstanceRequirements` de la solicitud en el mismo formato que utilizaría para crear o actualizar un grupo de escalado automático. Para obtener más información, consulte Vista previa de tipos de instancia con atributos especificados en la Guía del usuario de Amazon EC2.

Recursos relacionados

Para obtener más información sobre la selección del tipo de instancia basada en atributos, consulte Selección del tipo de instancia basada en atributos para EC2 Auto Scaling y EC2 Fleet en el blog. AWS

Puede declarar la selección del tipo de instancia basada en atributos al crear un grupo de escalado automático mediante CloudFormation. Para obtener más información, consulte el fragmento de ejemplo en la sección Ejemplos de plantillas de escalado automático en la Guía del usuario de CloudFormation .

Creación de un grupo de instancias mixtas seleccionando manualmente los tipos de instancias

En este tema, se muestra cómo lanzar varios tipos de instancia en un solo grupo de escalado automático eligiendo manualmente los tipos de instancia.

Si prefiere utilizar los atributos de instancia como criterios para seleccionar tipos de instancia, consulte Creación de grupos de instancias mixtas mediante la selección del tipo de instancia basada en atributos.

Contenido

- Requisitos previos
- Creación de un grupo de instancias mixtas (consola)
- Creación de un grupo de instancias mixtas (AWS CLI)
- Configuraciones de ejemplo

Requisitos previos

- Cree una plantilla de lanzamiento. Para obtener más información, consulte Creación de una plantilla de lanzamiento para un grupo de Auto Scaling.
- Verifique que la plantilla de lanzamiento no solicite ya instancias de spot.

Creación de un grupo de instancias mixtas (consola)

Para crear un grupo de instancias mixtas, elija manualmente qué tipos de instancias puede lanzar su grupo. Para ayudarlo a realizar los pasos de forma eficiente, se omiten algunas secciones opcionales.

Para revisar las prácticas recomendadas para un grupo de instancias mixtas, consulte Información general de la configuración de creación de un grupo de instancias mixtas.

Creación de un grupo de instancias mixtas

1. Abra la consola Amazon EC2 en https://console.aws.amazon.com/ec2/y seleccione Auto Scaling Groups en el panel de navegación.
2. En la barra de navegación de la parte superior de la pantalla, elija la misma Región de AWS que utilizó cuando creó la plantilla de lanzamiento.
3. Elija Create an Auto Scaling group (Crear un grupo de escalado automático).
4. En la página Choose launch template or configuration (Elegir una plantilla de lanzamiento o configuración), ingrese un nombre para el grupo de escalado automático.
5. Para elegir la plantilla de lanzamiento, haga lo siguiente:
    a. En launch template (Plantilla de lanzamiento), elija una plantilla de lanzamiento existente.
    b. Para Launch template version (Versión de plantilla de lanzamiento), decida si el grupo de escalado automático utiliza el valor predeterminado, la última versión o una versión específica de la plantilla de lanzamiento para escalado horizontal.
    c. Compruebe que la plantilla de lanzamiento admita todas las opciones que tenga previsto utilizar y, a continuación, elija Next (Siguiente).
6. En la página Elegir las opciones de lanzamiento de instancias, haga lo siguiente:
    a. En Instance type requirements (Requisitos de tipo de instancias), elija Override launch template (Anular la plantilla de lanzamiento) y, a continuación, elija Manually add instance types (Agregar los tipos de instancia de forma manual).

b. Elija los tipos de instancia. Puede utilizar nuestras recomendaciones como punto de partida. La opción Family and generation flexible (Familia y generación flexible) está seleccionada de forma predeterminada.

   - Para cambiar el orden de los tipos de instancia, utilice las flechas. Si elige una estrategia de asignación que admita la priorización, el orden de los tipos de instancia establece su prioridad de lanzamiento.
   - Para eliminar un tipo de instancia, elija X.
   - (Opcional) En los cuadros de la columna Ponderación, asigne a cada tipo de instancia una ponderación relativa. Para ello, ingrese el número de unidades que una instancia de ese tipo cuenta para alcanzar la capacidad deseada del grupo. Esto podría resultar útil si, por ejemplo, los tipos de instancia ofrecen diferentes funcionalidades de vCPU, memoria, almacenamiento o ancho de banda de la red. Para obtener más información, consulte Configuración de un grupo de escalado automático para utilizar ponderación de instancias.

     Tenga en cuenta que, si elige usar las recomendaciones de Tamaño flexible, todos los tipos de instancia que formen parte de esta sección tendrán automáticamente un valor de ponderación. Si no quiere especificar ningún peso, desactiva los cuadros de la columna Weight (Peso) para todos los tipos de instancia.

c. En Instance purchase options (Opciones de compra) para la distribución de instancias, especifique los porcentajes del grupo que se lanzará como instancias bajo demanda e instancias de spot, respectivamente. Si la suya es una aplicación sin estado, tolerante a errores y capaz de gestionar la interrupción de una instancia, puede especificar un mayor porcentaje de instancias de spot.

d. (Opcional) Cuando especifique un porcentaje para las instancias de spot, seleccione Incluir capacidad base bajo demanda y luego especifique la cantidad mínima de la capacidad inicial del grupo de escalado automático que deben satisfacer las instancias bajo demanda. Lo que esté más allá de la capacidad base utiliza la configuración de Instances distribution (Distribución de las instancias) para determinar cuántas instancias bajo demanda y de spot deben lanzarse.

e. En Allocation strategies (Estrategias de asignación), para On-Demand allocation strategy (Estrategia de asignación bajo demanda), elija una estrategia de asignación. Al elegir manualmente los tipos de instancia, se selecciona Prioritized (Priorizada) de forma predeterminada.

f. En Spot allocation strategy (Estrategia de asignación de spot), elija una estrategia de asignación. Price capacity optimized (Capacidad de precios optimizada) se selecciona de forma predeterminada.

   Si elige Capacidad optimizada, puede opcionalmente marcar la casilla Priorizar los tipos de instancia para que Amazon EC2 Auto Scaling pueda elegir qué tipo de instancia lanzar primero en función del orden en que aparecen los tipos de instancia.

g. En Reequilibrio de la capacidad, elija si desea habilitar o desactivar el reequilibrio de la capacidad. Use el reequilibrio de la capacidad para responder automáticamente cuando sus instancias de spot se aproximen a su finalización por una interrupción de spot. Para obtener más información, consulte Reequilibrio de la capacidad en Auto Scaling para reemplazar las instancias de spot en riesgo.

h. En Network (Red), para la opción VPC, elija una VPC. El grupo de escalado automático debe crearse en la misma VPC que el grupo de seguridad especificado en la plantilla de lanzamiento.

i. En Availability Zones and subnets (Zonas de disponibilidad y subredes), elija una o más subredes de la VPC especificada. Utilice subredes en varias zonas de disponibilidad para lograr una alta disponibilidad. Para obtener más información, consulte Consideraciones a la hora de elegir subredes de VPC.

j. Elija Siguiente, Siguiente.

7. Para el paso Configure group size and scaling policies (Configurar el tamaño del grupo y las políticas de escalado), haga lo siguiente:

   a. En Tamaño de grupo, para Capacidad deseada, introduzca el número inicial de instancias que desea lanzar.

      De forma predeterminada, la capacidad deseada se expresa como la cantidad de instancias. Si ha asignado ponderaciones a sus tipos de instancia, debe convertir este valor en la misma unidad de medida que utilizó para asignar las ponderaciones, por ejemplo, el número de vCPUs.

   b. En la sección Escalado, en Límites de escalado, si el nuevo valor de Capacidad deseada es superior a Capacidad deseada mínima y a Capacidad deseada máxima, Capacidad deseada máxima se incrementará automáticamente al nuevo valor de capacidad deseada. Puede cambiar estos límites según sea necesario. Para obtener más información, consulte Establecimiento de límites de escalado para el grupo de escalado automático.

8. Elija Skip to review (Omitir para revisar).

9. En la página Review (Revisar), elija Create Auto Scaling group (Crear grupo de escalado automático).

Creación de un grupo de instancias mixtas (AWS CLI)

Creación de un grupo de instancias mixtas mediante la línea de comandos

Utilice uno de los siguientes comandos:

- create-auto-scaling-group (AWS CLI)
- Nuevo- ASAuto ScalingGroup (AWS Tools for Windows PowerShell)

Configuraciones de ejemplo

En las siguientes configuraciones de ejemplo se muestra cómo crear grupos de instancias mixtas mediante las diferentes estrategias de asignación de spot.

> **Note**
>
> En estos ejemplos se muestra cómo utilizar un archivo de configuración con formato JSON o YAML. Si usa la AWS CLI versión 1, debe especificar un archivo de configuración con formato JSON. Si usa la AWS CLI versión 2, puede especificar un archivo de configuración formateado en YAML o JSON.

Ejemplos

- Ejemplo 1: Lanzamiento de Instancias de spot con la estrategia de asignación capacity-optimized
- Ejemplo 2: Lanzamiento de Instancias de spot con la estrategia de asignación capacity-optimized-prioritized
- Ejemplo 3: Lanzamiento de instancias de spot mediante la estrategia de asignación lowest-price diversificada en dos grupos
- Ejemplo 4: inicialización de instancias de spot con la estrategia de asignación price-capacity-optimized

Ejemplo 1: Lanzamiento de Instancias de spot con la estrategia de asignación **`capacity-optimized`**

El siguiente create-auto-scaling-groupcomando crea un grupo de Auto Scaling que especifica lo siguiente:

- El porcentaje del grupo que se va a lanzar como instancias bajo demanda (`0`) y un número base de instancias bajo demanda con el que se va a comenzar (1).
- Los tipos de instancia que se van a lanzar por orden de prioridad (`c5.large`, `c5a.large`, `m5.large`, `m5a.large`, `c4.large`, `m4.large`, `c3.large`, `m3.large`).
- Las subredes en las que se lanzarán las instancias (`subnet-5ea0c127`, `subnet-6194ea3b`, `subnet-c934b782`). Cada una corresponde a una zona de disponibilidad diferente.
- La plantilla de lanzamiento (`my-launch-template`) y la versión de la plantilla de lanzamiento (`$Default`).

Cuando Amazon EC2 Auto Scaling intenta cubrir su capacidad en diferido, lanza primero el tipo de instancia `c5.large`. Las instancias de spot proceden del grupo de spot óptimo en cada zona de disponibilidad en función de la capacidad de la instancia de spot.

JSON

```
aws autoscaling create-auto-scaling-group --cli-input-json file://~/config.json
```

El archivo `config.json` contiene la salida siguiente.

```
{
    "AutoScalingGroupName": "my-asg",
    "MixedInstancesPolicy": {
        "LaunchTemplate": {
            "LaunchTemplateSpecification": {
                "LaunchTemplateName": "my-launch-template",
                "Version": "$Default"
            },
            "Overrides": [
                {
                    "InstanceType": "c5.large"
                },
                {
                    "InstanceType": "c5a.large"
                },
```

```
                {
                    "InstanceType": "m5.large"
                },
                {
                    "InstanceType": "m5a.large"
                },
                {
                    "InstanceType": "c4.large"
                },
                {
                    "InstanceType": "m4.large"
                },
                {
                    "InstanceType": "c3.large"
                },
                {
                    "InstanceType": "m3.large"
                }
            ]
        },
        "InstancesDistribution": {
            "OnDemandBaseCapacity": 1,
            "OnDemandPercentageAboveBaseCapacity": 0,
            "SpotAllocationStrategy": "capacity-optimized"
        }
    },
    "MinSize": 1,
    "MaxSize": 5,
    "DesiredCapacity": 3,
    "VPCZoneIdentifier": "subnet-5ea0c127,subnet-6194ea3b,subnet-c934b782"
}
```

YAML

Como alternativa, puede usar el siguiente create-auto-scaling-groupcomando para crear el grupo Auto Scaling. Esto hace referencia a un archivo YAML como único parámetro de su grupo de escalado automático.

```
aws autoscaling create-auto-scaling-group --cli-input-yaml file://~/config.yaml
```

El archivo `config.yaml` contiene la salida siguiente.

```
---
```

```
AutoScalingGroupName: my-asg
MixedInstancesPolicy:
  LaunchTemplate:
    LaunchTemplateSpecification:
      LaunchTemplateName: my-launch-template
      Version: $Default
    Overrides:
    - InstanceType: c5.large
    - InstanceType: c5a.large
    - InstanceType: m5.large
    - InstanceType: m5a.large
    - InstanceType: c4.large
    - InstanceType: m4.large
    - InstanceType: c3.large
    - InstanceType: m3.large
  InstancesDistribution:
    OnDemandBaseCapacity: 1
    OnDemandPercentageAboveBaseCapacity: 0
    SpotAllocationStrategy: capacity-optimized
MinSize: 1
MaxSize: 5
DesiredCapacity: 3
VPCZoneIdentifier: subnet-5ea0c127,subnet-6194ea3b,subnet-c934b782
```

Ejemplo 2: Lanzamiento de Instancias de spot con la estrategia de asignación **`capacity-optimized-prioritized`**

El siguiente create-auto-scaling-groupcomando crea un grupo de Auto Scaling que especifica lo siguiente:

- El porcentaje del grupo que se va a lanzar como instancias bajo demanda (`0`) y un número base de instancias bajo demanda con el que se va a comenzar (`1`).
- Los tipos de instancia que se van a lanzar por orden de prioridad (`c5.large`, `c5a.large`, `m5.large`, `m5a.large`, `c4.large`, `m4.large`, `c3.large`, `m3.large`).
- Las subredes en las que se lanzarán las instancias (`subnet-5ea0c127`, `subnet-6194ea3b`, `subnet-c934b782`). Cada una corresponde a una zona de disponibilidad diferente.
- La plantilla de lanzamiento (`my-launch-template`) y la versión de la plantilla de lanzamiento (`$Latest`).

Cuando Amazon EC2 Auto Scaling intenta cubrir su capacidad en diferido, lanza primero el tipo de instancia `c5.large`. Cuando Amazon EC2 Auto Scaling intenta satisfacer la capacidad de spot, respeta las prioridades del tipo de instancia sobre la base del mejor esfuerzo. Sin embargo, primero optimiza la capacidad.

JSON

```
aws autoscaling create-auto-scaling-group --cli-input-json file://~/config.json
```

El archivo `config.json` contiene la salida siguiente.

```
{
    "AutoScalingGroupName": "my-asg",
    "MixedInstancesPolicy": {
        "LaunchTemplate": {
            "LaunchTemplateSpecification": {
                "LaunchTemplateName": "my-launch-template",
                "Version": "$Latest"
            },
            "Overrides": [
                {
                    "InstanceType": "c5.large"
                },
                {
                    "InstanceType": "c5a.large"
                },
                {
                    "InstanceType": "m5.large"
                },
                {
                    "InstanceType": "m5a.large"
                },
                {
                    "InstanceType": "c4.large"
                },
                {
                    "InstanceType": "m4.large"
                },
                {
                    "InstanceType": "c3.large"
                },
                {
                    "InstanceType": "m3.large"
```

```
                }
            ]
        },
        "InstancesDistribution": {
            "OnDemandBaseCapacity": 1,
            "OnDemandPercentageAboveBaseCapacity": 0,
            "SpotAllocationStrategy": "capacity-optimized-prioritized"
        }
    },
    "MinSize": 1,
    "MaxSize": 5,
    "DesiredCapacity": 3,
    "VPCZoneIdentifier": "subnet-5ea0c127,subnet-6194ea3b,subnet-c934b782"
}
```

YAML

Como alternativa, puede usar el siguiente create-auto-scaling-groupcomando para crear el grupo Auto Scaling. Esto hace referencia a un archivo YAML como único parámetro de su grupo de escalado automático.

```
aws autoscaling create-auto-scaling-group --cli-input-yaml file://~/config.yaml
```

El archivo `config.yaml` contiene la salida siguiente.

```
---
AutoScalingGroupName: my-asg
MixedInstancesPolicy:
  LaunchTemplate:
    LaunchTemplateSpecification:
      LaunchTemplateName: my-launch-template
      Version: $Default
    Overrides:
    - InstanceType: c5.large
    - InstanceType: c5a.large
    - InstanceType: m5.large
    - InstanceType: m5a.large
    - InstanceType: c4.large
    - InstanceType: m4.large
    - InstanceType: c3.large
    - InstanceType: m3.large
  InstancesDistribution:
    OnDemandBaseCapacity: 1
```

```
    OnDemandPercentageAboveBaseCapacity: 0
    SpotAllocationStrategy: capacity-optimized-prioritized
MinSize: 1
MaxSize: 5
DesiredCapacity: 3
VPCZoneIdentifier: subnet-5ea0c127,subnet-6194ea3b,subnet-c934b782
```

Ejemplo 3: Lanzamiento de instancias de spot mediante la estrategia de asignación **`lowest-price`** diversificada en dos grupos

El siguiente create-auto-scaling-groupcomando crea un grupo de Auto Scaling que especifica lo siguiente:

- El porcentaje del grupo que lanzar como instancias bajo demanda (`50`). (Esto no especifica un número base de instancias bajo demanda para empezar).
- Los tipos de instancia que se van a lanzar por orden de prioridad (`c5.large`, `c5a.large`, `m5.large`, `m5a.large`, `c4.large`, `m4.large`, `c3.large`, `m3.large`).
- Las subredes en las que se lanzarán las instancias (`subnet-5ea0c127`, `subnet-6194ea3b`, `subnet-c934b782`). Cada una corresponde a una zona de disponibilidad diferente.
- La plantilla de lanzamiento (`my-launch-template`) y la versión de la plantilla de lanzamiento (`$Latest`).

Cuando Amazon EC2 Auto Scaling intenta cubrir su capacidad en diferido, lanza primero el tipo de instancia `c5.large`. Para su capacidad de spot, Amazon EC2 Auto Scaling intenta lanzar las instancias de spot de manera uniforme en los dos grupos de precio más bajo de cada zona de disponibilidad.

JSON

```
aws autoscaling create-auto-scaling-group --cli-input-json file://~/config.json
```

El archivo `config.json` contiene la salida siguiente.

```
{
    "AutoScalingGroupName": "my-asg",
    "MixedInstancesPolicy": {
        "LaunchTemplate": {
            "LaunchTemplateSpecification": {
                "LaunchTemplateName": "my-launch-template",
```

```
                "Version": "$Latest"
            },
            "Overrides": [
                {
                    "InstanceType": "c5.large"
                },
                {
                    "InstanceType": "c5a.large"
                },
                {
                    "InstanceType": "m5.large"
                },
                {
                    "InstanceType": "m5a.large"
                },
                {
                    "InstanceType": "c4.large"
                },
                {
                    "InstanceType": "m4.large"
                },
                {
                    "InstanceType": "c3.large"
                },
                {
                    "InstanceType": "m3.large"
                }
            ]
        },
        "InstancesDistribution": {
            "OnDemandPercentageAboveBaseCapacity": 50,
            "SpotAllocationStrategy": "lowest-price",
            "SpotInstancePools": 2
        }
    },
    "MinSize": 1,
    "MaxSize": 5,
    "DesiredCapacity": 3,
    "VPCZoneIdentifier": "subnet-5ea0c127,subnet-6194ea3b,subnet-c934b782"
}
```

YAML

Como alternativa, puede usar el siguiente create-auto-scaling-groupcomando para crear el grupo Auto Scaling. Esto hace referencia a un archivo YAML como único parámetro de su grupo de escalado automático.

```
aws autoscaling create-auto-scaling-group --cli-input-yaml file://~/config.yaml
```

El archivo `config.yaml` contiene la salida siguiente.

```
---
AutoScalingGroupName: my-asg
MixedInstancesPolicy:
  LaunchTemplate:
    LaunchTemplateSpecification:
      LaunchTemplateName: my-launch-template
      Version: $Default
    Overrides:
    - InstanceType: c5.large
    - InstanceType: c5a.large
    - InstanceType: m5.large
    - InstanceType: m5a.large
    - InstanceType: c4.large
    - InstanceType: m4.large
    - InstanceType: c3.large
    - InstanceType: m3.large
  InstancesDistribution:
    OnDemandPercentageAboveBaseCapacity: 50
    SpotAllocationStrategy: lowest-price
    SpotInstancePools: 2
MinSize: 1
MaxSize: 5
DesiredCapacity: 3
VPCZoneIdentifier: subnet-5ea0c127,subnet-6194ea3b,subnet-c934b782
```

Ejemplo 4: inicialización de instancias de spot con la estrategia de asignación **`price-capacity-optimized`**

El siguiente create-auto-scaling-groupcomando crea un grupo de Auto Scaling que especifica lo siguiente:

- El porcentaje del grupo que lanzar como instancias bajo demanda (`30`). (Esto no especifica un número base de instancias bajo demanda para empezar).
- Los tipos de instancia que se van a lanzar por orden de prioridad (`c5.large`, `c5a.large`, `m5.large`, `m5a.large`, `c4.large`, `m4.large`, `c3.large`, `m3.large`).
- Las subredes en las que se lanzarán las instancias (`subnet-5ea0c127`, `subnet-6194ea3b`, `subnet-c934b782`). Cada una corresponde a una zona de disponibilidad diferente.
- La plantilla de lanzamiento (`my-launch-template`) y la versión de la plantilla de lanzamiento (`$Latest`).

Cuando Amazon EC2 Auto Scaling intenta cubrir su capacidad en diferido, lanza primero el tipo de instancia `c5.large`. Para su capacidad de spot, Amazon EC2 Auto Scaling intenta lanzar las instancias de spot desde grupos de instancias de spot con el precio más bajo posible, pero también con una capacidad óptima para la cantidad de instancias que lanza.

JSON

```
aws autoscaling create-auto-scaling-group --cli-input-json file://~/config.json
```

El archivo `config.json` contiene la salida siguiente.

```
{
    "AutoScalingGroupName": "my-asg",
    "MixedInstancesPolicy": {
        "LaunchTemplate": {
            "LaunchTemplateSpecification": {
                "LaunchTemplateName": "my-launch-template",
                "Version": "$Latest"
            },
            "Overrides": [
                {
                    "InstanceType": "c5.large"
                },
                {
                    "InstanceType": "c5a.large"
                },
                {
                    "InstanceType": "m5.large"
                },
                {
                    "InstanceType": "m5a.large"
```

```
                },
                {
                    "InstanceType": "c4.large"
                },
                {
                    "InstanceType": "m4.large"
                },
                {
                    "InstanceType": "c3.large"
                },
                {
                    "InstanceType": "m3.large"
                }
            ]
        },
        "InstancesDistribution": {
            "OnDemandPercentageAboveBaseCapacity": 30,
            "SpotAllocationStrategy": "price-capacity-optimized"
        }
    },
    "MinSize": 1,
    "MaxSize": 5,
    "DesiredCapacity": 3,
    "VPCZoneIdentifier": "subnet-5ea0c127,subnet-6194ea3b,subnet-c934b782"
}
```

YAML

Como alternativa, puede usar el siguiente create-auto-scaling-groupcomando para crear el grupo Auto Scaling. Esto hace referencia a un archivo YAML como único parámetro de su grupo de escalado automático.

```
aws autoscaling create-auto-scaling-group --cli-input-yaml file://~/config.yaml
```

El archivo `config.yaml` contiene la salida siguiente.

```
---
AutoScalingGroupName: my-asg
MixedInstancesPolicy:
  LaunchTemplate:
    LaunchTemplateSpecification:
      LaunchTemplateName: my-launch-template
```

```
      Version: $Default
    Overrides:
    - InstanceType: c5.large
    - InstanceType: c5a.large
    - InstanceType: m5.large
    - InstanceType: m5a.large
    - InstanceType: c4.large
    - InstanceType: m4.large
    - InstanceType: c3.large
    - InstanceType: m3.large
  InstancesDistribution:
    OnDemandPercentageAboveBaseCapacity: 30
    SpotAllocationStrategy: price-capacity-optimized
MinSize: 1
MaxSize: 5
DesiredCapacity: 3
VPCZoneIdentifier: subnet-5ea0c127,subnet-6194ea3b,subnet-c934b782
```

Configuración de un grupo de escalado automático para utilizar ponderación de instancias

Cuando utiliza varios tipos de instancias, puede especificar cuántas unidades desea asociar a cada tipo de instancia y, a continuación, especificar la capacidad del grupo con la misma unidad de medida. Esta opción de especificación de capacidad se conoce como ponderaciones.

Por ejemplo, supongamos que ejecuta una aplicación de procesamiento intensivo que funciona mejor con al menos 8 v CPUs y 15 GiB de RAM. Si utiliza `c5.2xlarge` como unidad base, cualquiera de los siguientes tipos de instancias EC2 satisfaría las necesidades de la aplicación.

Ejemplo de tipos de instancias

Tipo de instancia	vCPU	Memoria (GiB)
`c5.2xlarge`	8	16
`c5.4xlarge`	16	32
`c5.12xlarge`	48	96
`c5.18xlarge`	72	144
`c5.24xlarge`	96	192

De forma predeterminada, todos los tipos de instancias tienen la misma ponderación, independientemente de su tamaño. En otras palabras, tanto si Amazon EC2 Auto Scaling lanza un tipo de instancias grande como pequeño, todas las instancias cuentan a la hora de determinar la capacidad deseada del grupo de escalado automático.

Sin embargo, con las ponderaciones, se asigna un valor numérico que especifica cuántas unidades se van a asociar a cada tipo de instancia. Por ejemplo, si las instancias tienen diferentes tamaños, una instancia `c5.2xlarge` podría tener una ponderación de 2, mientras que una instancia `c5.4xlarge` (que es dos veces mayor) podría tener una ponderación de 4, etc. Luego, cuando Amazon EC2 Auto Scaling escala el grupo, estas ponderaciones se traducen en la cantidad de unidades que cada instancia tiene en cuenta para calcular la capacidad deseada.

Las ponderaciones no cambian los tipos de instancias que Amazon EC2 Auto Scaling decide lanzar; en su lugar, lo hacen las estrategias de asignación. Para obtener más información, consulte Estrategias de asignación de varios tipos de instancias.

> **Important**
>
> Para configurar un grupo de Auto Scaling para que alcance la capacidad deseada utilizando el número de v CPUs o la cantidad de memoria de cada tipo de instancia, recomendamos usar la selección del tipo de instancia basada en atributos. Al configurar el parámetro `DesiredCapacityType`, se especifica automáticamente el número de unidades que se van a asociar a cada tipo de instancia en función del valor que haya establecido para este parámetro. Para obtener más información, consulte Creación de grupos de instancias mixtas mediante la selección del tipo de instancia basada en atributos.

Contenido

- Consideraciones
- Comportamientos de ponderación de instancias
- Configuración de un grupo de escalado automático para utilizar ponderaciones
- Ejemplo de precio de spot por hora de unidad

Consideraciones

En esta sección, se analizan las consideraciones clave para implementar con eficacia las ponderaciones.

- Elija algunos tipos de instancias que se ajusten a las necesidades de rendimiento de su aplicación. Decida la ponderación con la que debe contar cada tipo de instancias en la capacidad deseada del grupo de escalado automático en función de sus capacidades. Estas ponderaciones se aplican a las instancias actuales y futuras.
- Evite intervalos amplios entre las ponderaciones. Por ejemplo, no especifique una ponderación de 1 para un tipo de instancia si el siguiente tipo de instancia mayor tiene una ponderación de 200. La diferencia entre las ponderaciones más bajas y más altas tampoco debe ser exagerada. Las diferencias extremas de ponderación pueden afectar negativamente a la optimización de la relación costo-rendimiento.
- Especifique la capacidad deseada del grupo en unidades, no en instancias. Por ejemplo, si las ponderaciones se basan en la vCPU, establezca el número deseado de núcleos que quiere utilizar, así como el número mínimo y máximo.
- Establezca las ponderaciones y la capacidad deseada para que esta sea al menos dos o tres veces mayor que su ponderación más alta.

Tenga en cuenta lo siguiente al actualizar los grupos existentes:

- Cuando añada ponderaciones a un grupo existente, incluya las ponderaciones de todos los tipos de instancias que se utilizan actualmente.
- Cuando añada o cambie ponderaciones existentes, Amazon EC2 Auto Scaling inicializará o terminará las instancias para alcanzar su capacidad deseada en función de los nuevos valores de ponderación.
- Si quita un tipo de instancia, las instancias en ejecución de ese tipo seguirán teniendo la última ponderación, aunque ya no estén definidas.

Comportamientos de ponderación de instancias

Cuando utiliza ponderaciones de instancias, Amazon EC2 Auto Scaling se comporta de la siguiente manera:

- La capacidad actual será igual o superior a la capacidad deseada. La capacidad actual puede superar la capacidad deseada si las instancias inicializadas superan las unidades de capacidad deseadas restantes. Por ejemplo, supongamos que especifica dos tipos de instancias: `c5.2xlarge` y `c5.12xlarge`, y que asigna la ponderación 2 a `c5.2xlarge` y la ponderación 12 a `c5.12xlarge`. Si faltan cinco unidades para satisfacer la capacidad deseada, y Amazon

EC2 Auto Scaling aprovisiona una `c5.12xlarge`, la capacidad deseada se sobrepasa en siete unidades.

- Al inicializar instancias, Amazon EC2 Auto Scaling prioriza la distribución de la capacidad entre las zonas de disponibilidad y el respeto de las estrategias de asignación en lugar de superar la capacidad deseada.
- Amazon EC2 Auto Scaling puede superar el límite máximo de capacidad para mantener el equilibrio entre las zonas de disponibilidad con las estrategias de asignación que desee. El límite máximo exigido por Amazon EC2 Auto Scaling es su capacidad deseada más la ponderación más alta.

Configuración de un grupo de escalado automático para utilizar ponderaciones

Puede configurar un grupo de escalado automático para usar ponderaciones, como se muestra en los siguientes ejemplos de AWS CLI . Para obtener instrucciones sobre cómo utilizar la consola, consulte Creación de un grupo de instancias mixtas seleccionando manualmente los tipos de instancias.

Configuración de un nuevo grupo de escalado automático para utilizar ponderaciones (AWS CLI)

Utilice el comando create-auto-scaling-group. Por ejemplo, el comando siguiente crea un nuevo grupo de escalado automático y asigna ponderaciones al especificar lo siguiente:

- El porcentaje del grupo que lanzar como instancias en diferido (`0`)
- La estrategia de asignación de instancias de spot de cada zona de disponibilidad (`capacity-optimized`)
- Los tipos de instancia que se van a lanzar por orden de prioridad (`m4.16xlarge`, `m5.24xlarge`)
- Los pesos de las instancias que corresponden a la diferencia de tamaño relativo (vCPUs) entre los tipos de instancias (16,) 24
- Las subredes en las que lanzar las instancias (`subnet-5ea0c127`, `subnet-c934b782`)`subnet-6194ea3b`, cada una de ellas correspondiente a una zona de disponibilidad diferente
- La plantilla de lanzamiento (`my-launch-template`) y la versión de la plantilla de lanzamiento (`$Latest`)

```
aws autoscaling create-auto-scaling-group --cli-input-json file://~/config.json
```

El archivo `config.json` contiene la salida siguiente.

```
{
    "AutoScalingGroupName": "my-asg",
    "MixedInstancesPolicy": {
        "LaunchTemplate": {
            "LaunchTemplateSpecification": {
                "LaunchTemplateName": "my-launch-template",
                "Version": "$Latest"
            },
            "Overrides": [
                {
                    "InstanceType": "m4.16xlarge",
                    "WeightedCapacity": "16"
                },
                {
                    "InstanceType": "m5.24xlarge",
                    "WeightedCapacity": "24"
                }
            ]
        },
        "InstancesDistribution": {
            "OnDemandPercentageAboveBaseCapacity": 0,
            "SpotAllocationStrategy": "capacity-optimized"
        }
    },
    "MinSize": 160,
    "MaxSize": 720,
    "DesiredCapacity": 480,
    "VPCZoneIdentifier": "subnet-5ea0c127,subnet-6194ea3b,subnet-c934b782",
    "Tags": []
}
```

Configuración de un grupo de escalado automático existente para utilizar ponderaciones (AWS CLI)

Utilice el comando update-auto-scaling-group. Por ejemplo, el comando siguiente asigna ponderaciones a los tipos de instancias de un grupo de escalado automático existente al especificar lo siguiente:

- Los tipos de instancia que se van a lanzar por orden de prioridad (`c5.18xlarge`, `c5.24xlarge`, `c5.2xlarge`, `c5.4xlarge`)

- Los pesos de las instancias que corresponden a la diferencia de tamaño relativo (vCPUs) entre los tipos de instancias (18,24,2,4)
- La nueva capacidad deseada, que es mayor que la ponderación más alta

```
aws autoscaling update-auto-scaling-group --cli-input-json file://~/config.json
```

El archivo `config.json` contiene la salida siguiente.

```
{
    "AutoScalingGroupName": "my-existing-asg",
    "MixedInstancesPolicy": {
        "LaunchTemplate": {
            "Overrides": [
                {
                    "InstanceType": "c5.18xlarge",
                    "WeightedCapacity": "18"
                },
                {
                    "InstanceType": "c5.24xlarge",
                    "WeightedCapacity": "24"
                },
                {
                    "InstanceType": "c5.2xlarge",
                    "WeightedCapacity": "2"
                },
                {
                    "InstanceType": "c5.4xlarge",
                    "WeightedCapacity": "4"
                }
            ]
        }
    },
    "MinSize": 0,
    "MaxSize": 100,
    "DesiredCapacity": 100
}
```

Verificación de las ponderaciones con la línea de comandos

Utilice uno de los siguientes comandos:

- describe-auto-scaling-groups (AWS CLI)
- Obtenga- ASAuto ScalingGroup (AWS Tools for Windows PowerShell)

Ejemplo de precio de spot por hora de unidad

En la siguiente tabla, se compara el precio por hora de las instancias de spot en diferentes zonas de disponibilidad del Este de EE. UU. (Norte de Virginia) con el precio de las instancias bajo demanda de la misma región. Los precios mostrados son solo ejemplos; no son los precios actuales. Estos son sus costos por hora de instancia.

Ejemplo: Precio de las instancias de spot por hora de instancia

Tipo de instancia	us-east-1a	us-east-1b	us-east-1c	Precios bajo demanda
c5.2xlarge	0,180 USD	0,191 USD	0,170 USD	0,34 USD
c5.4xlarge	0,341 USD	0,361 USD	0,318 USD	0,68 USD
c5.12xlarge	0,779 USD	0,777 USD	0,777 USD	2,04 USD
c5.18xlarge	1,207 USD	1,475 USD	1,357 USD	3,06 USD
c5.24xlarge	1,555 USD	1,555 USD	1,555 USD	4,08 USD

Con las ponderaciones de instancias, puede evaluar los costos en función del uso por hora de unidad. Puede calcular el precio por hora de unidad dividiendo el precio de un tipo de instancia por el número de unidades que representa. En las instancias bajo demanda, el precio por hora de unidad es el mismo cuando se implementa un solo tipo de instancia y cuando se implementa un tamaño diferente del mismo tipo de instancia. Sin embargo, el precio de spot por hora de unidad varía en función del grupo de spot.

En el siguiente ejemplo, se muestra cómo funciona el cálculo del precio de spot por hora de unidad con las ponderaciones de las instancias. Para facilitar el cálculo, supongamos que desea lanzar instancias de spot solo en `us-east-1a`. El precio por hora de unidad se muestra en la siguiente tabla.

Ejemplo: precio de spot por hora de unidad

Tipo de instancia	us-east-1a	Ponderación de instancia	Precio por hora de unidad
`c5.2xlarge`	0,180 USD	2	0,090 USD
`c5.4xlarge`	0,341 USD	4	0,085 USD
`c5.12xlarge`	0,779 USD	12	0,065 USD
`c5.18xlarge`	1,207 USD	18	0,067 USD
`c5.24xlarge`	1,555 USD	24	0,065 USD

Utilización de plantillas de lanzamiento múltiples

Además de utilizar varios tipos de instancia, también puede utilizar varias plantillas de lanzamiento.

Por ejemplo, supongamos que configura un grupo de escalado automático para aplicaciones de uso informático intensivo y desea incluir una combinación de tipos de instancias C5, C5a y C6g. Sin embargo, las instancias C6g cuentan con un procesador AWS Graviton basado en la arquitectura Arm de 64 bits, mientras que las instancias C5 y C5a funcionan con procesadores Intel x86 de 64 bits. Tanto AMIs las instancias C5 como las C5a funcionan en cada una de esas instancias, pero no en las instancias C6g. Para solucionar este problema, use una plantilla de lanzamiento diferente para las instancias de C6g. Puede seguir utilizando la misma plantilla de lanzamiento para las instancias C5 y C5a.

Esta sección contiene los procedimientos que se utilizan para realizar tareas relacionadas con el AWS CLI uso de varias plantillas de lanzamiento. Actualmente, esta característica solo está disponible si utiliza la AWS CLI o un SDK, y no está disponible desde la consola.

Contenido

- Configuración de un grupo de escalado automático para utilizar varias plantillas de lanzamiento

- Recursos relacionados

Configuración de un grupo de escalado automático para utilizar varias plantillas de lanzamiento

Puede configurar un grupo de escalado automático para que utilice varias plantillas de lanzamiento, como se muestra en los siguientes ejemplos.

Configuración de un nuevo grupo de escalado automático para que utilice varias plantillas de lanzamiento (AWS CLI)

Utilice el comando create-auto-scaling-group. Por ejemplo, el comando siguiente crea un nuevo grupo de escalado automático. Especifica los tipos de instancias `c5.large`, `c5a.large` y `c6g.large`, y define una nueva plantilla de lanzamiento para el tipo de instancias `c6g.large`, para asegurarse de que se utiliza una AMI adecuada para lanzar instancias de ARM. Amazon EC2 Auto Scaling utiliza el orden de los tipos de instancias para determinar qué tipo de instancias se utilizará en primer lugar al cumplir con la capacidad en diferido.

```
aws autoscaling create-auto-scaling-group --cli-input-json file://~/config.json
```

El archivo `config.json` contiene la salida siguiente.

```
{
  "AutoScalingGroupName":"my-asg",
  "MixedInstancesPolicy":{
    "LaunchTemplate":{
      "LaunchTemplateSpecification":{
        "LaunchTemplateName":"my-launch-template-for-x86",
        "Version":"$Latest"
      },
      "Overrides":[
        {
          "InstanceType":"c6g.large",
          "LaunchTemplateSpecification": {
            "LaunchTemplateName": "my-launch-template-for-arm",
            "Version": "$Latest"
          }
        },
        {
          "InstanceType":"c5.large"
        },
        {
```

```
          "InstanceType":"c5a.large"
        }
      ]
    },
    "InstancesDistribution":{
      "OnDemandBaseCapacity": 1,
      "OnDemandPercentageAboveBaseCapacity": 50,
      "SpotAllocationStrategy": "capacity-optimized"
    }
  },
  "MinSize":1,
  "MaxSize":5,
  "DesiredCapacity":3,
  "VPCZoneIdentifier":"subnet-5ea0c127,subnet-6194ea3b,subnet-c934b782",
  "Tags":[ ]
}
```

Configuración de un grupo de escalado automático existente para que utilice varias plantillas de lanzamiento (AWS CLI)

Utilice el comando update-auto-scaling-group. Por ejemplo, el siguiente comando asigna la plantilla de lanzamiento nombrada *my-launch-template-for-arm* al tipo de *c6g.large* instancia del grupo de escalado automático denominado *my-asg*.

```
aws autoscaling update-auto-scaling-group --cli-input-json file://~/config.json
```

El archivo `config.json` contiene la salida siguiente.

```
{
  "AutoScalingGroupName":"my-asg",
  "MixedInstancesPolicy":{
    "LaunchTemplate":{
      "Overrides":[
        {
          "InstanceType":"c6g.large",
          "LaunchTemplateSpecification": {
            "LaunchTemplateName": "my-launch-template-for-arm",
            "Version": "$Latest"
          }
        },
        {
          "InstanceType":"c5.large"
```

```
        },
        {
          "InstanceType":"c5a.large"
        }
      ]
    }
  }
}
```

Cómo configurar un nuevo grupo de escalado automático para utilizar varias plantillas de lanzamiento con una selección del tipo de instancia basada en atributos (AWS CLI)

Utilice el comando create-auto-scaling-group. Por ejemplo, el siguiente comando crea un nuevo grupo de Auto Scaling especificando una plantilla de lanzamiento para las instancias de AWS Graviton con una AMI ARM y una plantilla de lanzamiento adicional para las instancias basadas en AMD o Intel con una AMI x86. Luego utiliza dos veces la selección de instancias basada en atributos para seleccionar entre un rango amplio de tipos de instancia para cada arquitectura de CPU. Puede añadir una configuración similar a un grupo de Auto Scaling existente con el update-autoscaling-groupcomando.

```
aws autoscaling create-auto-scaling-group --cli-input-json file://~/config.json
```

El archivo `config.json` contiene la salida siguiente.

```
{
  "AutoScalingGroupName":"my-asg",
  "MixedInstancesPolicy":{
    "LaunchTemplate":{
      "LaunchTemplateSpecification":{
        "LaunchTemplateName":"my-launch-template-for-arm",
        "Version":"$Latest"
      },
      "Overrides":[
        {
          "InstanceRequirements": {
            "VCpuCount": {"Min": 2},
            "MemoryMiB": {"Min": 2048},
            "CpuManufacturers": ["amazon-web-services"]
          }
        },
        {
          "InstanceRequirements": {
```

```
            "VCpuCount": {"Min": 2},
            "MemoryMiB": {"Min": 2048},
            "CpuManufacturers": ["intel", "amd"]
          },
          "LaunchTemplateSpecification": {
            "LaunchTemplateName": "my-launch-template-for-x86",
            "Version": "$Latest"
          }
        }
      ]
    },
    "InstancesDistribution":{
      "OnDemandPercentageAboveBaseCapacity": 0,
      "SpotAllocationStrategy": "price-capacity-optimized"
    }
  },
  "MinSize":1,
  "MaxSize":10,
  "DesiredCapacity":6,
  "VPCZoneIdentifier":"subnet-5ea0c127,subnet-6194ea3b,subnet-c934b782",
  "Tags":[ ]
}
```

Para verificar las plantillas de lanzamiento para un grupo de Auto Scaling

Utilice uno de los siguientes comandos:

- describe-auto-scaling-groups (AWS CLI)
- Get- ASAuto ScalingGroup (AWS Tools for Windows PowerShell)

Recursos relacionados

Puedes encontrar un ejemplo de cómo especificar varias plantillas de lanzamiento mediante la selección del tipo de instancia basada en atributos en una CloudFormation plantilla en AWS Re:post.

Creación de grupos de escalado automático mediante configuraciones de lanzamiento

Important

Limitaciones:

- Desde el 1 de enero de 2023, los nuevos tipos de instancias de Amazon EC2 ya no son compatibles con las configuraciones de lanzamiento. Esto incluye la compatibilidad con cualquier tipo de instancia que se añada y Región de AWS después del lanzamiento inicial de la región.
- Las cuentas creadas el 1 de junio de 2023, o después de dicha fecha, no pueden crear nuevas configuraciones de lanzamiento con la consola.
- Las cuentas creadas a partir del 1 de octubre de 2024 no pueden crear nuevas configuraciones de lanzamiento mediante ningún método (consola AWS CLI, API o CloudFormation).

Migre a plantillas de inicialización para asegurarse de no tener que crear nuevas configuraciones de inicialización ahora o en el futuro. Para obtener información sobre la migración de sus grupos de escalado automático a plantillas de lanzamiento, consulte Migración de sus grupos de escalado automático a plantillas de lanzamiento.

Si ha creado una configuración de lanzamiento o una instancia de EC2, puede crear un grupo de escalado automático que use una configuración de lanzamiento como plantilla de configuración para sus instancias de EC2. La configuración de lanzamiento especifica información como el ID de AMI, el tipo de instancia, el par de claves, los grupos de seguridad y la asignación de dispositivos de bloques para las instancias. Para obtener información sobre la creación de configuraciones de lanzamiento, consulte Crear una configuración de lanzamiento.

Usted debe tener los permisos suficientes para poder crear un grupo de escalado automático. También debe tener permisos suficientes para crear el rol vinculado a servicio que Amazon EC2 Auto Scaling utiliza para realizar acciones en su nombre si este no existe todavía. Para ver ejemplos de políticas de IAM que un administrador puede utilizar como referencia para concederle permisos, consulte Ejemplos de políticas basadas en identidades.

Contenido

- Crear un grupo de Auto Scaling mediante una configuración de lanzamiento
- Cree un grupo de Auto Scaling a partir de una instancia existente mediante el AWS CLI

Crear un grupo de Auto Scaling mediante una configuración de lanzamiento

> **Important**
>
> Proporcionamos información sobre las configuraciones de lanzamiento para los clientes que aún no han migrado las configuraciones de lanzamiento a las plantillas de lanzamiento. Para obtener información sobre la migración de sus grupos de escalado automático a plantillas de lanzamiento, consulte Migración de sus grupos de escalado automático a plantillas de lanzamiento.

Cuando crea un grupo de escalado automático, debe especificar la información necesaria para configurar las instancias de Amazon EC2, las zonas de disponibilidad y las subredes de VPC para las instancias, la capacidad deseada y los límites de capacidad mínima y máxima.

El siguiente procedimiento demuestra cómo crear un grupo de Auto Scaling mediante una configuración de lanzamiento. No puede modificar una configuración de lanzamiento después de crearla, pero puede sustituir la configuración de lanzamiento por un grupo de Auto Scaling. Para obtener más información, consulte Cambio en la configuración de lanzamiento de un grupo de escalado automático.

Requisitos previos

- Tiene que haber creado una configuración de lanzamiento. Para obtener más información, consulte Crear una configuración de lanzamiento.

Para crear un grupo de Auto Scaling mediante una configuración de lanzamiento (consola)

1. Abra la consola Amazon EC2 en https://console.aws.amazon.com/ec2/y seleccione Auto Scaling Groups en el panel de navegación.
2. En la barra de navegación situada en la parte superior de la pantalla, elija la misma Región de AWS que utilizó al crear la configuración de lanzamiento.
3. Elija Create an Auto Scaling group (Crear un grupo de escalado automático).
4. En la página Choose launch template or configuration (Elegir una plantilla de lanzamiento o configuración), ingrese un nombre para el grupo de Auto Scaling.
5. Para elegir una configuración de lanzamiento, haga lo siguiente:

a. En Launch template (Plantilla de lanzamiento), elija Switch to launch configuration (Cambiar a configuración de lanzamiento).

b. En Launch configuration (Configuración de lanzamiento), elija una configuración de lanzamiento existente.

c. Compruebe que la configuración de lanzamiento admita todas las opciones que tiene previsto utilizar y, a continuación, elija Next (Siguiente).

6. En la página Configurar las opciones de lanzamiento de las instancias, en Red, para la opción VPC, elija una VPC. El grupo de Auto Scaling debe crearse en la misma VPC que el grupo de seguridad especificado en la configuración de lanzamiento.
7. En Zonas de disponibilidad y subredes, elija una o más subredes de la VPC especificada. Utilice subredes en varias zonas de disponibilidad para lograr una alta disponibilidad. Para obtener más información, consulte Consideraciones a la hora de elegir subredes de VPC.
8. Elija Siguiente.

   O puede aceptar el resto de las opciones predeterminadas y elegir Skip to review (Omitir para revisar).
9. (Opcional) En la página Configure advanced options (Configurar opciones avanzadas) configure las siguientes opciones y, a continuación, elija Next (Siguiente):

   a. (Opcional) En Comprobaciones de estado, Tipos de comprobaciones de estado adicionales, seleccione Activar las comprobaciones de estado de Amazon EBS. Para obtener más información, consulte Supervisión de las instancias de escalado automático con volúmenes de Amazon EBS dañados mediante comprobaciones de estado.

   b. (Opcional) En Período de gracia de comprobación de estado, ingrese el tiempo, en segundos. Este es la cantidad de tiempo que Amazon EC2 Auto Scaling debe esperar antes de comprobar el estado de una instancia una vez que pasa al estado `InService`. Para obtener más información, consulte Establezca el periodo de gracia de la comprobación de estado para un grupo de escalado automático.

   c. En Configuración adicional, Supervisión, elija si desea habilitar la recopilación de métricas CloudWatch grupales. Estas métricas proporcionan mediciones que pueden ser indicadores de un posible problema, como la cantidad de instancias en proceso de terminación o la cantidad de instancias pendientes. Para obtener más información, consulte Supervise CloudWatch las métricas de sus grupos e instancias de Auto Scaling.

   d. En Habilitar la preparación predeterminada de instancias, seleccione esta opción y elija el tiempo de preparación de la aplicación. Si está creando un grupo de Auto Scaling que tiene

una política de escalado, la función de calentamiento de instancias predeterminada mejora CloudWatch las métricas de Amazon utilizadas para el escalado dinámico. Para obtener más información, consulte Establecimiento de la preparación predeterminada de instancias para un grupo de escalado automático.

10. (Opcional) En la página Configure group size and scaling policies (Configurar políticas de tamaño de grupo y escala) configure las siguientes opciones y, a continuación, elija Next (Siguiente):

    a. En Tamaño de grupo, para Capacidad deseada, introduzca el número inicial de instancias que desea inicializar.

    b. En la sección Escalado, en Límites de escalado, si el nuevo valor de Capacidad deseada es superior a Capacidad deseada mínima y a Capacidad deseada máxima, Capacidad deseada máxima se incrementará automáticamente al nuevo valor de capacidad deseada. Puede cambiar estos límites según sea necesario. Para obtener más información, consulte Establecimiento de límites de escalado para el grupo de escalado automático.

    c. En Escalado automático, elija si desea crear una política de escalado de seguimiento de destino. También puede crear esta política después de crear su grupo de escalado automático.

       Si elige Política de escalado de seguimiento de destino, siga las instrucciones en Creación de una política de escalado de seguimiento de destino para crear la política.

    d. En Política de mantenimiento de instancias, elija si desea crear una política de mantenimiento de instancias. También puede crear esta política después de crear su grupo de escalado automático. Siga las instrucciones de Establecimiento de una política de mantenimiento de instancias para crear la política.

    e. En Protección de reducción horizontal de instancias, elija si desea habilitar la protección de reducción horizontal de instancias. Para obtener más información, consulte Uso de la protección frente a la reducción horizontal de instancias para controlar la terminación de instancias.

11. (Opcional) Para recibir notificaciones, en Add notification (Añadir notificación), configure la notificación y, a continuación, elija Next (Siguiente). Para obtener más información, consulte Opciones de notificaciones de Amazon SNS para Amazon EC2 Auto Scaling.

12. (Opcional) Para añadir etiquetas, elija Add Tags (Añadir etiquetas), facilite un valor y una clave de etiqueta, y luego elija Next (Siguiente). Para obtener más información, consulte Etiquetado de grupos e instancias de Auto Scaling.

13. En la página Review (Revisar), elija Create Auto Scaling group (Crear grupo de escalado automático).

Para crear un grupo de Auto Scaling mediante la línea de comandos

Puede utilizar uno de los siguientes comandos:

- create-auto-scaling-group (AWS CLI)
- Nuevo- () ASAuto ScalingGroupAWS Tools for Windows PowerShell

Cree un grupo de Auto Scaling a partir de una instancia existente mediante el AWS CLI

> **Important**
>
> Proporcionamos información sobre las configuraciones de lanzamiento para los clientes que aún no han migrado las configuraciones de lanzamiento a las plantillas de lanzamiento. Para obtener información sobre la migración de sus grupos de escalado automático a plantillas de lanzamiento, consulte Migración de sus grupos de escalado automático a plantillas de lanzamiento.

Si es la primera vez que crea un grupo de Auto Scaling, se recomienda utilizar la consola para crear una plantilla de lanzamiento a partir de una instancia de EC2 ya existente. A continuación, utilice la plantilla de lanzamiento para crear un nuevo grupo de Auto Scaling. Para informarse sobre este procedimiento, consulte Creación de un grupo de Auto Scaling mediante el asistente de lanzamiento de Amazon EC2.

En el siguiente procedimiento se muestra cómo crear un grupo de escalado automático mediante la especificación de una instancia ya existente que se utilizará como base para lanzar otras instancias. Para crear una instancia de EC2 son necesarios varios parámetros, como el ID de la imagen de Amazon Machine (AMI), el tipo de instancia, el par de claves y el grupo de seguridad. Amazon EC2 Auto Scaling también utiliza toda esta información para lanzar instancias en su nombre cuando sea necesario escalar. Esta información se almacena en una plantilla de lanzamiento o una configuración de lanzamiento.

Cuando se utiliza una instancia ya existente, Amazon EC2 Auto Scaling crea un grupo de escalado automático que lanza instancias según una configuración de lanzamiento que se crea al mismo tiempo. La nueva configuración de lanzamiento recibe el mismo nombre que el grupo de Auto Scaling e incluye determinados detalles de configuración de la instancia identificada.

Los siguientes detalles de configuración se copian de la instancia identificada en la configuración de lanzamiento:

- ID DE AMI
- Tipo de instancia
- Par de claves
- Grupos de seguridad
- Tipo de dirección IP (pública o privada)
- Perfil de instancias de IAM, si corresponde
- Supervisión (verdadero o falso)
- EBS optimizado (verdadero o falso)
- Ajustes de tenencia, si se inicia en una VPC (compartida o dedicada)
- ID del kernel e ID del disco RAM, si procede
- Datos del usuario, si se especifican
- Precio (máximo) de spot

La subred de VPC y la zona de disponibilidad se copian de la instancia identificada a la propia definición de recursos del grupo de escalado automático.

Si la instancia identificada pertenece a un grupo de ubicación, el nuevo grupo de Auto Scaling lanza instancias en el mismo grupo de ubicación que la instancia identificada. Dado que los ajustes de configuración de lanzamiento no permiten especificar un grupo de ubicación, este se copia en el atributo `PlacementGroup` del nuevo grupo de Auto Scaling.

Los siguientes detalles de configuración no se copian de la instancia identificada:

- Almacenamiento: los dispositivos de bloques (volúmenes de EBS y volúmenes de almacén de instancias) no se copian de la instancia identificada. Por el contrario, la asignación de dispositivos de bloques creada durante la creación de la AMI determina qué dispositivos se utilizan.

- Número de interfaces de red: las interfaces de red no se copian de la instancia identificada. Por el contrario, Amazon EC2 Auto Scaling utiliza su configuración predeterminada para crear una interfaz de red, que es la de la red primaria (eth0).
- Opciones de metadatos de la instancia: las configuraciones del límite de saltos de la respuesta del token, de la versión de los metadatos y de los metadatos accesibles no se copian de la instancia identificada. Por el contrario, Amazon EC2 Auto Scaling utiliza su configuración predeterminada. Para obtener más información, consulte Configurar las opciones de metadatos de instancia.
- Equilibradores de carga: si la instancia identificada se registra con uno o más equilibradores de carga, la información relativa a ellos no se copia en el atributo del grupo de destino o del equilibrador de carga del nuevo grupo de Auto Scaling.
- Etiquetas: si la instancia identificada tiene etiquetas, estas no se copian en el atributo `Tags` del nuevo grupo de Auto Scaling.

Requisitos previos

La instancia EC2 debe cumplir los siguientes criterios:

- La instancia no es miembro de otro grupo de Auto Scaling.
- La instancia tiene el estado `running`.
- La AMI que se utilizó para lanzar la instancia debe existir.

Cree un grupo de Auto Scaling desde una instancia EC2 (AWS CLI)

El siguiente procedimiento muestra cómo utilizar un comando CLI para crear un grupo de escalado automático a partir de una instancia EC2.

Este procedimiento no agrega la instancia al grupo de Auto Scaling. Para asociar la instancia, debe ejecutar el comando attach-instances una vez creado el grupo de escalado automático.

Antes de comenzar, busque el ID de la instancia de EC2 mediante la consola de Amazon EC2 o el comando describe-instances.

Para usar la instancia actual como plantilla

- Utilice el siguiente create-auto-scaling-groupcomando para crear un grupo de Auto Scalingmy-asg-from-instance, desde la instancia `i-123456789abcdefg0` EC2.

```
aws autoscaling create-auto-scaling-group --auto-scaling-group-name my-asg-from-instance \
  --instance-id i-123456789abcdefg0 --min-size 1 --max-size 2 --desired-capacity 2
```

Para verificar que el grupo de Auto Scaling ha lanzado instancias

- Utilice el siguiente describe-auto-scaling-groupscomando para comprobar que el grupo Auto Scaling se creó correctamente.

```
aws autoscaling describe-auto-scaling-groups --auto-scaling-group-name my-asg-from-instance
```

La siguiente respuesta de ejemplo muestra que la capacidad de respuesta deseada del grupo es 2, que el grupo dispone de 2 instancias de ejecución y que la configuración de lanzamiento también se denomina `my-asg-from-instance`.

```
{
  "AutoScalingGroups":[
    {
      "AutoScalingGroupName":"my-asg-from-instance",
      "AutoScalingGroupARN":"arn",
      "LaunchConfigurationName":"my-asg-from-instance",
      "MinSize":1,
      "MaxSize":2,
      "DesiredCapacity":2,
      "DefaultCooldown":300,
      "AvailabilityZones":[
        "us-west-2a"
      ],
      "LoadBalancerNames":[],
      "TargetGroupARNs":[],
      "HealthCheckType":"EC2",
      "HealthCheckGracePeriod":0,
      "Instances":[
        {
          "InstanceId":"i-34567890abcdef012",
          "InstanceType":"t2.micro",
          "AvailabilityZone":"us-west-2a",
          "LifecycleState":"InService",
          "HealthStatus":"Healthy",
```

```
          "LaunchConfigurationName":"my-asg-from-instance",
          "ProtectedFromScaleIn":false
        },
        {
          "InstanceId":"i-012345abcdefg6789",
          "InstanceType":"t2.micro",
          "AvailabilityZone":"us-west-2a",
          "LifecycleState":"InService",
          "HealthStatus":"Healthy",
          "LaunchConfigurationName":"my-asg-from-instance",
          "ProtectedFromScaleIn":false
        }
      ],
      "CreatedTime":"2020-10-28T02:39:22.152Z",
      "SuspendedProcesses":[ ],
      "VPCZoneIdentifier":"subnet-0abc1234",
      "EnabledMetrics":[ ],
      "Tags":[ ],
      "TerminationPolicies":[
        "Default"
      ],
      "NewInstancesProtectedFromScaleIn":false,
      "ServiceLinkedRoleARN":"arn",
      "TrafficSources":[]
    }
  ]
}
```

Para ver la configuración de lanzamiento

- Utilice el siguiente describe-launch-configurationscomando para ver los detalles de la configuración de lanzamiento.

  ```
  aws autoscaling describe-launch-configurations --launch-configuration-names my-asg-from-instance
  ```

  A continuación, se muestra un ejemplo de la salida:

  ```
  {
    "LaunchConfigurations":[
      {
  ```

```
      "LaunchConfigurationName":"my-asg-from-instance",
      "LaunchConfigurationARN":"arn",
      "ImageId":"ami-234567890abcdefgh",
      "KeyName":"my-key-pair-uswest2",
      "SecurityGroups":[
        "sg-12abcdefgh3456789"
      ],
      "ClassicLinkVPCSecurityGroups":[ ],
      "UserData":"",
      "InstanceType":"t2.micro",
      "KernelId":"",
      "RamdiskId":"",
      "BlockDeviceMappings":[ ],
      "InstanceMonitoring":{
        "Enabled":true
      },
      "CreatedTime":"2020-10-28T02:39:22.321Z",
      "EbsOptimized":false,
      "AssociatePublicIpAddress":true
    }
  ]
}
```

Para terminar la instancia

- Puede terminar la instancia si ya no la necesita. El siguiente comando terminate-instances termina la instancia `i-123456789abcdefg0`.

  ```
  aws ec2 terminate-instances --instance-ids i-123456789abcdefg0
  ```

  Después de terminar una instancia de Amazon EC2, no puede reiniciarla. Al terminar un volumen, sus datos se pierden y el volumen no se puede adjuntar a ninguna instancia. Para obtener más información sobre la terminación de instancias, consulte Terminar una instancia en la Guía del usuario de Amazon EC2.

Lanza instancias de forma sincrónica

Amazon EC2 Auto Scaling proporciona dos métodos para lanzar instancias en su grupo de Auto Scaling: comportamiento de escalado asíncrono y aprovisionamiento sincrónico mediante la API. LaunchInstances

Con el aprovisionamiento sincrónico, utiliza la LaunchInstances API para solicitar un número específico de instancias en una zona de disponibilidad concreta. El aprovisionamiento sincrónico ofrece las siguientes ventajas:

- Información inmediata sobre la disponibilidad de la capacidad en zonas de disponibilidad específicas
- Control preciso sobre las instancias de la zona de disponibilidad que se inician
- Identificadores de instancia deterministas para su uso inmediato en sistemas de orquestación
- Real-time escalar las decisiones en función de las limitaciones de capacidad reales
- Escalado más rápido al eliminar los tiempos de espera para los lanzamientos asíncronos de Auto Scaling

Con el Auto Scaling asíncrono, cuando cambia la capacidad deseada o cuando se activa una política de escalado, Amazon EC2 Auto Scaling procesa la solicitud de escalado y lanza las instancias en segundo plano. Debe supervisar las actividades de escalado o describir su grupo de Auto Scaling para determinar cuándo las instancias se han lanzado correctamente.

> **Note**
>
> - La LaunchInstances API solo funciona con grupos de Auto Scaling que utilizan plantillas de lanzamiento. No se admiten los grupos de Auto Scaling que utilizan configuraciones de lanzamiento. Si su grupo de Auto Scaling usa una configuración de lanzamiento, debe migrar a una plantilla de lanzamiento antes de usar el aprovisionamiento sincrónico.
> - La LaunchInstances API admite políticas de instancias mixtas con opciones de compra total On-Demand o totalmente puntual únicamente. No se admiten políticas mixtas que combinen ambas On-Demand instancias con instancias puntuales.
> - Para los grupos de Auto Scaling que cubren varias zonas de disponibilidad, debe especificar la zona de disponibilidad o subred de destino. Para los grupos Single-AZ, este parámetro es opcional.

Aprovisionamiento sincrónico y escalado asíncrono

Aprovisionamiento sincrónico

Cuando utiliza la LaunchInstances API, Amazon EC2 Auto Scaling:

- Intenta lanzar inmediatamente las instancias solicitadas mediante CreateFleet
- Espera a que se devuelvan CreateFleet los ID de las instancias antes de responder
- Devuelve los ID de instancia, los tipos de instancia y la información sobre las zonas de disponibilidad en caso de éxito
- Devuelve códigos de error específicos y detalles sobre el error
- Proporciona información inmediata, lo que permite tomar decisiones de escalado en tiempo real

Escalado asíncrono

Cuando utiliza métodos de Auto Scaling asíncronos, como cambiar la capacidad deseada o utilizar políticas de escalado, Amazon EC2 Auto Scaling:

- Actualiza la capacidad deseada en la API, pero no devuelve las instancias inmediatamente
- Planifica automáticamente el lanzamiento de las instancias en todas las zonas de disponibilidad
- Lanza instancias mediante flujos de trabajo en segundo plano
- Distribuye automáticamente la capacidad entre varias zonas de disponibilidad para lograr un equilibrio
- Gestiona los errores de lanzamiento con una lógica de reintento integrada

Debe sondear las actividades de escalado o describir su grupo de Auto Scaling para comprobar el estado de las operaciones de lanzamiento.

Limitaciones y consideraciones

Cuando trabaje con el aprovisionamiento sincrónico, tenga en cuenta las siguientes notas y limitaciones:

- Estado de la instancia tras el lanzamiento: las instancias devueltas por la API están en estado pendiente. Es posible que sigan fallando durante los procesos de flujo de trabajo o los enlaces del ciclo de vida posteriores. Una respuesta correcta de la API significa que EC2 ha aceptado la solicitud de lanzamiento y ha devuelto los ID de las instancias. Las instancias no se consideran

automáticamente completamente listas para las cargas de trabajo y deben completar los procesos de ciclo de vida estándar de EC2 y Auto Scaling.

- Limitación de piscinas calientes: actualmente no se admiten grupos de Auto Scaling con piscinas calientes. Si intenta llamar a la LaunchInstances API en un grupo de Auto Scaling que tiene una piscina caliente configurada, la API realiza un arranque en frío en lugar de utilizar instancias de piscina caliente y devuelve un UnsupportedOperation error. Para obtener más información sobre los arranques en frío, consulte Limitaciones de las piscinas calientes.
- Tiempo de espera de la API y reintentos: si la CreateFleet operación subyacente tarda más de lo esperado, es posible que la API agote el tiempo de espera y devuelva un token de idempotencia. Puedes volver ClientToken a intentarlo para hacer un seguimiento de la operación de lanzamiento original o usar describe-instances con el token del cliente para comprobar las instancias lanzadas.
- Restricciones de la zona de disponibilidad: si su grupo de Auto Scaling abarca varias zonas de disponibilidad y tiene habilitado el reequilibrio de zonas de disponibilidad, lanzar instancias de forma sincrónica puede provocar conflictos operativos:
  - Limitación de zona de disponibilidad única por llamada: cada llamada a la LaunchInstances API solo puede dirigirse a una zona de disponibilidad, incluso si su grupo de Auto Scaling abarca varias zonas.
  - Conflictos de reequilibrio de AZ: si tu grupo de Auto Scaling tiene activado el reequilibrio de AZ, las llamadas secuenciales entre diferentes AZ pueden activar más lanzamientos asíncronos, lo que resulta en más instancias de las previstas. Considere la posibilidad de suspender el reequilibrio de AZ para controlar la capacidad con precisión. Para obtener más información, consulte Suspensión y reanudación de procesos de Amazon EC2 Auto Scaling.
- Escenarios de éxito parcial: es posible que la `LaunchInstances` API devuelva un éxito parcial si solo está disponible parte de la capacidad solicitada, lo que es un comportamiento normal de EC2. La API devuelve las instancias lanzadas correctamente junto con los detalles de los errores de los lanzamientos fallidos. En los casos de uso que requieren que todas las instancias se lancen juntas (por ejemplo, las aplicaciones que necesitan que todas las instancias estén en la misma zona de disponibilidad para obtener una latencia baja), tendrás que cerrar las instancias lanzadas parcialmente y volver a intentarlo en una zona de disponibilidad diferente. Tenga en cuenta este comportamiento al diseñar la lógica de reintentos para cargas de trabajo sensibles a la capacidad.
- Ponderaciones de instancias: si su grupo de Auto Scaling usa ponderaciones de instancias, el RequestedCapacity parámetro representa las unidades de capacidad ponderadas, no el número de instancias. La cantidad real de instancias lanzadas depende de los tipos de instancias seleccionados y de sus pesos configurados. Auto Scaling de EC2 limita los lanzamientos a 100 instancias por llamada a la API, independientemente de la capacidad ponderada solicitada.

- Tipos de instancias mixtas: la LaunchInstances API usa la política de instancias mixtas existente de su grupo de Auto Scaling para determinar qué tipos de instancias lanzar. La API lanza las instancias de acuerdo con la estrategia de asignación y las prioridades de los tipos de instancias del grupo.

Lanzamiento de instancias con aprovisionamiento sincrónico

Puede usar la LaunchInstances API para lanzar sincrónicamente un número específico de instancias en su grupo de Auto Scaling. La API lanza las instancias en la zona de disponibilidad o subred que especifique y devuelve inmediatamente los ID de las instancias o la información de error.

Requisitos previos

Para poder usar la LaunchInstances API, debes tener:

- Un grupo de Auto Scaling que usa una plantilla de lanzamiento (no se admiten las configuraciones de lanzamiento)
- Debe tener permisos para las siguientes acciones de IAM:
  - `autoscaling:LaunchInstances`
  - `ec2:CreateFleet`
  - `ec2:DescribeLaunchTemplateVersions`

Lance instancias con aprovisionamiento sincrónico

Puede lanzar instancias con aprovisionamiento sincrónico a través de. AWS CLI

AWS CLI

Para lanzar instancias con aprovisionamiento sincrónico:

```
aws autoscaling launch-instances \
        --auto-scaling-group-name group-name \
        --requested-capacity number \
        [--availability-zones zone-name] \
        [--subnet-ids subnet-id] \
        [--availability-zone-ids zone-id] \
        [--retry-strategy none|retry-with-group-configuration] \
        [--client-token token]
```

Ejemplos

Lanzar instancias en una zona de disponibilidad específica

```
aws autoscaling launch-instances \
        --auto-scaling-group-name my-asg \
        --requested-capacity 3 \
        --availability-zones us-east-1a \
        --retry-strategy retry-with-group-configuration
```

Lanzar instancias en una subred específica

```
aws autoscaling launch-instances \
        --auto-scaling-group-name my-asg \
        --requested-capacity 2 \
        --subnet-ids subnet-12345678 \
        --retry-strategy none \
        --client-token my-unique-token-123
```

Gestionar las respuestas

Ejemplo de una respuesta exitosa:

```
{
    "AutoScalingGroupName": "my-asg",
    "ClientToken": "my-unique-token-123",
    "Instances": [
        {
            "InstanceType": "m5.xlarge",
            "AvailabilityZone": "us-east-1a",
            "AvailabilityZoneId": "use1-az1",
            "SubnetId": "subnet-12345678",
            "MarketType": "OnDemand",
            "InstanceIds": ["i-0123456789abcdef0", "i-0fedcba9876543210"]
        }
    ],
    "Errors": []
}
```

Ejemplo de una respuesta con errores

```
{
    "AutoScalingGroupName": "my-asg",
    "ClientToken": "my-unique-token-123",
    "Instances": [],
    "Errors": [
       {
        "InstanceType": "m5.large",
        "AvailabilityZone": "us-east-1a",
        "AvailabilityZoneId": "use1-az1",
        "SubnetId": "subnet-12345678",
        "MarketType": "OnDemand",
        "ErrorCode": "InsufficientInstanceCapacity",
        "ErrorMessage": "There is not enough capacity to fulfill your request for
 instance type 'm5.large' in 'us-east-1a'"
        }
    ]
}
```


Gestione los errores de lanzamiento y los reintentos

Cuando la LaunchInstances API detecte errores, puedes implementar estrategias de reintentos utilizando los tokens de idempotencia y las políticas de reintentos adecuadas.

Puedes usar el parámetro client-token para reintentar las solicitudes. También puedes usar las siguientes estrategias de reintento:

- `RetryStrategy: none`(predeterminado): si la llamada a la API falla, la capacidad deseada del grupo de Auto Scaling permanece sin cambios y no se produce ningún reintento automático.
- `RetryStrategy: retry-with-group-configuration`- Si la llamada a la API falla, la capacidad deseada del grupo de Auto Scaling aumentará en la cantidad solicitada y Auto Scaling volverá a intentar lanzar instancias automáticamente utilizando la configuración y los procesos estándar del grupo.

El comportamiento de los reintentos `RetryStrategy: retry-with-group-configuration` depende del tipo de error:

- Errores de validación: la capacidad deseada no se ha incrementado porque la operación no puede continuar. Por ejemplo, parámetros no válidos o configuraciones no compatibles.

- Errores de capacidad: se ha aumentado la capacidad deseada y Auto Scaling volverá a intentar lanzar las instancias de forma asíncrona mediante los procesos de escalado normales del grupo.

Uso de fichas de cliente para la idempotencia

El `client-token` parámetro garantiza las operaciones idempotentes y permite reintentos seguros de las solicitudes de lanzamiento.

Comportamientos clave:

- Los tokens de los clientes tienen una vida útil de 8 horas a partir de la solicitud inicial
- Al volver a intentarlo con el mismo token de cliente en un plazo de 8 horas, se devuelve la respuesta en caché en lugar de lanzar nuevas instancias
- Transcurridas 8 horas, el mismo token de cliente iniciará una nueva operación de lanzamiento

Update an Auto Scaling group

Puede actualizar la mayoría de los detalles de sus grupos de escalado automático. No puede actualizar el nombre de un grupo de Auto Scaling ni cambiarlo Región de AWS.

Actualización de un grupo de escalado automático (consola)

1. Abra la consola Amazon EC2 en https://console.aws.amazon.com/ec2/y seleccione Auto Scaling Groups en el panel de navegación.
2. Elija su grupo de escalado automático para mostrar información sobre el grupo, con pestañas para Detalles, Actividad, Escalado automático, Administración de instancias, Monitoreo y Actualización de instancias.
3. Elija las pestañas de las áreas de configuración que le interesen y actualice la configuración según sea necesario. Para cada ajuste que edite, elija Actualizar para guardar los cambios en la configuración del grupo de escalado automático.

    - Pestaña Detalles

      Estas son las configuraciones generales para su grupo de escalado automático. Puede editarlos y administrarlos de la misma manera que durante la creación del grupo de escalado automático.

La sección Configuraciones avanzadas tiene algunas opciones que no están disponibles al crear el grupo, como las políticas de terminación, la recuperación, los procesos suspendidos y la duración máxima de la instancia. También puede ver, pero no editar, el grupo de ubicación y el rol vinculado al servicio del grupo de escalado automático.

- Pestaña integraciones
  - Equilibrio de carga: Elastic Load Balancing

    Si el grupo está asociado a los recursos de Elastic Load Balancing, consulte Agregar una zona de disponibilidad antes de cambiar las zonas de disponibilidad. Es posible que algunas restricciones del equilibrador de carga le impidan aplicar los cambios en las zonas de disponibilidad de su grupo a las zonas de disponibilidad de su equilibrador de carga.
  - Opciones de integración de VPC Lattice: VPC Lattice
  - Cambio de zona ARC: Cambio de zona del grupo de escalado automático
- Pestaña Escalado automático
  - Políticas de escalado dinámico: políticas de escalado dinámico
  - Políticas de escalado predictivo: políticas de escalado predictivo
  - Acciones programadas: acciones programadas
- Pestaña Administración de instancias
  - Enlaces de ciclo de vida: enlaces de ciclo de vida
  - Grupo en caliente: grupos en caliente
- Pestaña Actividad
  - Notificaciones de actividad: notificaciones de Amazon SNS
- Pestaña Monitoreo
  - Solo hay una opción en esta pestaña, que le permite habilitar o deshabilitar la recopilación de métricas CloudWatch grupales.

Actualización de un grupo de escalado automático mediante la línea de comandos

Puede utilizar uno de los siguientes comandos:

- update-auto-scaling-group (AWS CLI)
- Update-ASAutoScalingGroup (AWS Tools for Windows PowerShell)

Actualización de las instancias de escalado automático

Si asocia una nueva plantilla de lanzamiento o configuración de lanzamiento a un grupo de escalado automático, todas las instancias nuevas obtendrán la configuración actualizada. Las instancias existentes siguen ejecutándose con la configuración con la que se lanzaron la primera vez. Para aplicar sus cambios a las instancias existentes, tiene las siguientes opciones:

- Iniciar una actualización de instancias para reemplazar las instancias anteriores. Para obtener más información, consulte Uso de una actualización de instancias para actualizar instancias en un grupo de escalado automático.
- Aguardar a que las actividades de escalado reemplacen gradualmente las instancias más antiguas por las instancias más recientes en función de sus políticas de terminación.
- Terminarlas manualmente para que sean reemplazadas por su grupo de escalado automático.

> **Note**
>
> Puede cambiar los siguientes atributos de instancia especificándolos como parte de la plantilla de lanzamiento o de la configuración de lanzamiento:
>
> - Imagen de máquina de Amazon (AMI)
> - dispositivos de bloques
> - par de claves
> - tipo de instancia
> - grupos de seguridad
> - datos de usuario
> - supervisión
> - Perfil de instancia IAM
> - propiedad de ubicación
> - kernel
> - disco RAM
> - si la instancia tiene o no una dirección IP pública
> - la estrategia de distribución de la zona de disponibilidad

Estrategia de asignación del grupo de escalado automático y cambios en la capacidad

Al cambiar una estrategia de asignación de grupos de Auto Scaling, las instancias existentes no se reemplazarán. Todas las instancias nuevas que se lancen debido a eventos de escalado horizontal seguirán la nueva estrategia de asignación. Cualquier evento de reducción horizontal futuro seguirá la política de rescisión y utilizará la nueva estrategia de asignación si la política de terminación está establecida en `Default` o `AllocationStrategy`. Por ejemplo, si cambia la estrategia de asignación de `lowest-price` a `price-capacity-optimized`, es posible que no se termine ninguna instancia, pero se lanzará cualquier instancia nueva con la nueva estrategia de asignación. Los cambios en el tipo de instancia no afectan a las instancias existentes.

Al cambiar ciertos parámetros, como el OnDemandBaseCapacityo el OnDemandPercentageAboveBaseCapacity, Auto Scaling se reequilibrará automáticamente si el porcentaje de On-Demand instancias e instancias puntuales no coinciden con las nuevas especificaciones. Por ejemplo, supongamos que un grupo de Auto Scaling tiene `OnDemandPercentageAboveBaseCapacity` establecido un 50 por ciento de On-Demand instancias y un 50 por ciento de instancias puntuales. A continuación, `OnDemandPercentageAboveBaseCapacity` se aumenta al 100 por ciento de On-Demand instancias. El grupo Auto Scaling se reequilibrará de forma proactiva mediante el lanzamiento de nuevas On-Demand instancias y la cancelación de las instancias puntuales. La política de mantenimiento de instancias que usted definió determina el orden de las actividades de lanzamiento y terminación.

Etiquetado de grupos e instancias de Auto Scaling

Una etiqueta es una etiqueta de atributo personalizada que se asigna o que se asigna a AWS un recurso. AWS Cada etiqueta de tiene dos partes:

- Una clave de etiqueta (por ejemplo, `costcenter`, `environment` o `project`)
- Un campo opcional denominado valor de etiqueta (por ejemplo, `111122223333` o `production`)

Las etiquetas lo ayudan a hacer lo siguiente:

- Realiza un seguimiento de tus AWS costes. Estas etiquetas se activan en el Administración de facturación y costos de AWS panel de control. AWS utiliza las etiquetas para clasificar los costes

y entregarle un informe mensual de asignación de costes. Para obtener más información, consulte Uso de etiquetas de asignación de costos en la Guía del usuario de AWS Billing .

- Controle el acceso a grupos de Auto Scaling en función de las etiquetas. Puede utilizar condiciones en sus políticas de IAM para controlar el acceso a los grupos de Auto Scaling en función de las etiquetas de ese grupo. Para obtener más información, consulte Etiquetas para seguridad.
- Filtre y busque los grupos de escalado automático en función de las etiquetas que agregue. Para obtener más información, consulte Uso de etiquetas para filtrar grupos de Auto Scaling.
- Identifique y organice sus AWS recursos. Muchos Servicios de AWS admiten el etiquetado, por lo que puede asignar la misma etiqueta a los recursos de diferentes servicios para indicar que los recursos están relacionados.

Puede etiquetar grupos de Auto Scaling nuevos o existentes. También puede propagar etiquetas de un grupo de escalado automático a las instancias de Amazon EC2 que lance.

Las etiquetas no se propagan a volúmenes de Amazon EBS. Para agregar etiquetas a volúmenes de Amazon EBS, especifique las etiquetas en una plantilla de lanzamiento. Para obtener más información, consulte Creación de una plantilla de lanzamiento para un grupo de Auto Scaling.

Puedes crear y administrar etiquetas a través de los Consola de administración de AWS AWS CLI, o los SDK.

Contenido

- Restricciones de nombres y uso de las etiquetas
- Ciclo de vida de etiquetado de las instancias EC2
- Etiqueta los grupos de Auto Scaling
- Eliminar etiquetas
- Etiquetas para seguridad
- Control del acceso a las etiquetas
- Uso de etiquetas para filtrar grupos de Auto Scaling

Restricciones de nombres y uso de las etiquetas

Se aplican las siguientes restricciones básicas a las etiquetas de :

- El número máximo de etiquetas por recurso es 50.

- El número máximo de etiquetas que puede añadir o eliminar con una sola llamada es 25.
- La longitud máxima de la clave es de 128 caracteres Unicode.
- La longitud máxima del valor es de 256 caracteres Unicode.
- Las claves y los valores de las etiquetas distinguen entre mayúsculas y minúsculas. Como práctica recomendada, decida una estrategia de uso de mayúsculas y minúsculas en las etiquetas e implemente esa estrategia sistemáticamente en todos los tipos de recursos.
- No utilice el `aws:` prefijo en los nombres o valores de las etiquetas, ya que está reservado para AWS su uso. No puede editar ni eliminar nombres o valores de etiqueta con este prefijo, y no se tienen en cuenta para su cuota de etiquetas por recurso.

Ciclo de vida de etiquetado de las instancias EC2

Si ha decidido propagar las etiquetas a las instancias EC2, las etiquetas se administran de la siguiente manera:

- Cuando un grupo de Auto Scaling lanza instancias, agrega etiquetas a las instancias durante la creación de recursos y no después de haberlos creado.
- El grupo de Auto Scaling agrega automáticamente una etiqueta a las instancias con una clave de `aws:autoscaling:groupName` y un valor del nombre del grupo de Auto Scaling.
- Si especifica etiquetas de instancia en la plantilla de lanzamiento y opta por propagar las etiquetas del grupo a sus instancias, todas las etiquetas se fusionarán. Si se especifica la misma clave de etiqueta para una etiqueta en su plantilla de lanzamiento y una etiqueta en su grupo de Auto Scaling, entonces el valor de la etiqueta del grupo tiene prioridad.
- Cuando adjunta instancias existentes, el grupo de Auto Scaling agrega las etiquetas a las instancias, sobrescribiendo las etiquetas existentes con la misma clave de etiqueta. Asimismo, agrega una etiqueta con una clave de `aws:autoscaling:groupName` y un valor del nombre del grupo de Auto Scaling.
- Cuando desvincula una instancia de un grupo de Auto Scaling, solo se elimina la etiqueta `aws:autoscaling:groupName`.

Etiqueta los grupos de Auto Scaling

Cuando agrega una etiqueta a su grupo de Auto Scaling, puede especificar si se debe añadir a las instancias lanzadas en el grupo de Auto Scaling. Si modifica una etiqueta, la versión actualizada de la etiqueta se añade a las instancias lanzadas en el grupo de Auto Scaling tras el cambio. Si crea o

modifica una etiqueta para un grupo de Auto Scaling, estos cambios no se realizan en las instancias que ya se están ejecutando en el grupo de Auto Scaling.

Contenido

- Agregar o modificar etiquetas (consola)
- Agregue o modifique etiquetas (AWS CLI)

Agregar o modificar etiquetas (consola)

Para etiquetar un grupo de Auto Scaling en el momento de su creación

Si utiliza la consola de Amazon EC2 para crear un grupo de Auto Scaling, puede especificar claves y valores de etiqueta en la página Add tags (Agregar etiquetas) del asistente de creación de un grupo de Auto Scaling. Para propagar una etiqueta por las instancias lanzadas en el grupo de Auto Scaling, no olvide mantener la opción Tag New Instances (Etiquetar nuevas instancias) de dicha etiqueta seleccionada. En caso contrario, puede anular la selección.

Para agregar o modificar etiquetas en un grupo de Auto Scaling existente

1. Abra la consola Amazon EC2 en https://console.aws.amazon.com/ec2/y seleccione Auto Scaling Groups en el panel de navegación.
2. Seleccione la casilla situada junto al grupo de escalado automático.

   Se abre un panel dividido en la parte inferior de la página Auto Scaling groups (Grupos de Auto Scaling).
3. En la pestaña Details (Detalles), elija Tags (Etiquetas), Edit (Editar).
4. Para modificar las etiquetas existentes, edite Key y Value.
5. Para agregar una nueva etiqueta, seleccione Add tag y modifique Key y Value. Puede mantener la opción Tag new instances (Etiquetar instancias nuevas) seleccionada para agregar la etiqueta a las instancias lanzadas en el grupo de Auto Scaling automáticamente y cancelar su selección en caso contrario.
6. Cuando haya terminado de agregar etiquetas, elija Update (Actualizar).

Agregue o modifique etiquetas (AWS CLI)

Los siguientes ejemplos muestran cómo utilizarla para añadir etiquetas AWS CLI al crear grupos de Auto Scaling y cómo añadir o modificar etiquetas para los grupos de Auto Scaling existentes.

Para etiquetar un grupo de Auto Scaling en el momento de su creación

Utilice el comando create-auto-scaling-group para crear un nuevo grupo de Auto Scaling y agregar una etiqueta, por ejemplo, **`environment=production`**, en el grupo de Auto Scaling. La etiqueta también se añade a cualquier instancia lanzada en el grupo de Auto Scaling.

```
aws autoscaling create-auto-scaling-group --auto-scaling-group-name my-asg \
  --launch-configuration-name my-launch-config --min-size 1 --max-size 3 \
  --vpc-zone-identifier "subnet-5ea0c127,subnet-6194ea3b,subnet-c934b782" \
  --tags Key=environment,Value=production,PropagateAtLaunch=true
```

Para crear o modificar etiquetas en un grupo de Auto Scaling existente

Utilice el comando create-or-update-tags para crear o modificar una etiqueta. Por ejemplo, el siguiente comando añade las etiquetas **`costcenter=cc123`** y **`Name=my-asg`**. Las etiquetas también se agregarán a cualquier instancia que se lance en el grupo de Auto Scaling después de realizar este cambio. Si ya existe una etiqueta con esta clave, se sustituye la etiqueta existente. La consola de Amazon EC2 asocia el nombre mostrado de cada instancia con el nombre que especificado en la clave `Name` (distingue entre mayúsculas y minúsculas).

```
aws autoscaling create-or-update-tags \
  --tags ResourceId=my-asg,ResourceType=auto-scaling-group,Key=Name,Value=my-
asg,PropagateAtLaunch=true \
  ResourceId=my-asg,ResourceType=auto-scaling-
group,Key=costcenter,Value=cc123,PropagateAtLaunch=true
```

Describa las etiquetas de un grupo de Auto Scaling (AWS CLI)

Si desea ver las etiquetas que se aplican a un grupo de Auto Scaling específico, puede utilizar cualquiera de los siguientes comandos:

- describe-tags: proporcione el nombre de grupo de escalado automático para ver una lista de etiquetas del grupo especificado.

  ```
  aws autoscaling describe-tags --filters Name=auto-scaling-group,Values=my-asg
  ```

  A continuación, se muestra un ejemplo de respuesta.

  ```
  {
      "Tags": [
  ```

```
        {
            "ResourceType": "auto-scaling-group",
            "ResourceId": "my-asg",
            "PropagateAtLaunch": true,
            "Value": "production",
            "Key": "environment"
        }
    ]
}
```

- describe-auto-scaling-groups: proporcione el nombre del grupo de escalado automático para ver los atributos del grupo especificado, incluidas las etiquetas.

```
aws autoscaling describe-auto-scaling-groups --auto-scaling-group-name my-asg
```

A continuación, se muestra un ejemplo de respuesta.

```
{
    "AutoScalingGroups": [
        {
            "AutoScalingGroupName": "my-asg",
            "AutoScalingGroupARN": "arn",
            "LaunchTemplate": {
                "LaunchTemplateId": "lt-0b97f1e282EXAMPLE",
                "LaunchTemplateName": "my-launch-template",
                "Version": "$Latest"
            },
            "MinSize": 1,
            "MaxSize": 5,
            "DesiredCapacity": 1,
            ...
            "Tags": [
                {
                    "ResourceType": "auto-scaling-group",
                    "ResourceId": "my-asg",
                    "PropagateAtLaunch": true,
                    "Value": "production",
                    "Key": "environment"
                }
            ],
            ...
```

```
        }
    ]
}
```

Eliminar etiquetas

Puede eliminar una etiqueta asociada a su grupo de Auto Scaling en cualquier momento.

Contenido

- Eliminar etiquetas (consola)
- Eliminar etiquetas (AWS CLI)

Eliminar etiquetas (consola)

Cómo eliminar una etiqueta

1. Abra la consola Amazon EC2 en https://console.aws.amazon.com/ec2/y seleccione Auto Scaling Groups en el panel de navegación.
2. Seleccione la casilla de verificación situada junto a un grupo existente.

   Se abre un panel dividido en la parte inferior de la página Auto Scaling groups (Grupos de Auto Scaling).
3. En la pestaña Details (Detalles), elija Tags (Etiquetas), Edit (Editar).
4. Elija Remove (Eliminar) junto a la etiqueta.
5. Elija Actualizar.

Eliminar etiquetas (AWS CLI)

Ejecute el comando delete-tags para eliminar una etiqueta. Por ejemplo, el comando siguiente elimina una etiqueta cuya clave es **`environment`**.

```
aws autoscaling delete-tags --tags "ResourceId=my-asg,ResourceType=auto-scaling-
group,Key=environment"
```

Debe especificar la clave de etiqueta, pero no obligatorio especificar el valor. Si especifica un valor y el valor es incorrecto, la etiqueta no se elimina.

Etiquetas para seguridad

Utilice las etiquetas para comprobar que el solicitante (como un rol o usuario de IAM) tiene permisos para crear, modificar o eliminar grupos específicos de escalado automático. Proporcione información de etiquetas en el elemento de condición de una política de IAM mediante una o más de las siguientes claves de condición:

- Utilice `autoscaling:ResourceTag/`*`tag-key`*`: `*`tag-value`* para permitir (o denegar) acciones de los usuarios en grupos de Auto Scaling con etiquetas específicas.
- Utilice `aws:RequestTag/`*`tag-key`*`: `*`tag-value`* para exigir que una etiqueta específica esté presente o no en una solicitud.
- Utilice `aws:TagKeys [`*`tag-key`*`, ...]` para exigir que las claves de etiqueta específicas estén presentes o no en una solicitud.

Por ejemplo, puede denegar el acceso a todos los grupos de Auto Scaling que incluyan una etiqueta con la clave **`environment`** y el valor **`production`**, como se muestra en el ejemplo siguiente.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": [
                "autoscaling:CreateAutoScalingGroup",
                "autoscaling:UpdateAutoScalingGroup",
                "autoscaling:DeleteAutoScalingGroup"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {"autoscaling:ResourceTag/environment":
 "production"}
            }
        }
    ]
}
```

Para obtener más información acerca del uso de claves de condición para controlar el acceso a grupos de escalado automático, consulte Cómo funciona Amazon EC2 Auto Scaling con IAM.

Control del acceso a las etiquetas

Utilice las etiquetas para comprobar que el solicitante (como un rol o usuario de IAM) tiene permisos para agregar, modificar o eliminar etiquetas específicas de escalado automático.

El siguiente ejemplo de política de IAM da permiso a la entidad principal para eliminar sólo la etiqueta con la clave **`temporary`** de los grupos de escalado automático.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "autoscaling:DeleteTags",
            "Resource": "*",
            "Condition": {
                "ForAllValues:StringEquals": { "aws:TagKeys": ["temporary"] }
            }
        }
    ]
}
```

Para ver más ejemplos de políticas de IAM que imponen restricciones a las etiquetas especificadas para los grupos de escalado automático, consulte Controlar qué claves de etiqueta y valores de etiqueta se pueden utilizar.

> **Note**
>
> Aunque tenga una política que impida que los usuarios puedan realizar una operación de etiquetado (o eliminación de etiquetas) en un grupo de Auto Scaling, esto no les impedirá cambiar manualmente las etiquetas en las instancias una vez lanzadas. Para obtener ejemplos de control de acceso a las etiquetas en instancias de EC2, consulte Ejemplo: Etiquetar recursos en la Guía del usuario de Amazon EC2.

Uso de etiquetas para filtrar grupos de Auto Scaling

En los siguientes ejemplos se muestra cómo utilizar filtros con el comando describe-auto-scaling-groups para describir los grupos de Auto Scaling con etiquetas específicas. El filtrado por etiquetas se limita al SDK AWS CLI o a un SDK y no está disponible en la consola.

Consideraciones de filtrado

- Puede especificar varios filtros y varios valores de filtro en una sola solicitud.
- No puede utilizar comodines con los valores del filtro.
- Los valores de filtro distinguen entre mayúsculas y minúsculas.

Ejemplo: describir grupos de Auto Scaling con una clave de etiqueta y un par de valores específicos

En el siguiente comando se muestra cómo filtrar los resultados para que aparezcan únicamente los grupos de Auto Scaling con la clave de etiqueta y el par de valores de **`environment=production`**.

```
aws autoscaling describe-auto-scaling-groups \
  --filters Name=tag-key,Values=environment Name=tag-value,Values=production
```

A continuación, se muestra un ejemplo de respuesta.

```
{
    "AutoScalingGroups": [
        {
            "AutoScalingGroupName": "my-asg",
            "AutoScalingGroupARN": "arn",
            "LaunchTemplate": {
                "LaunchTemplateId": "lt-0b97f1e282EXAMPLE",
                "LaunchTemplateName": "my-launch-template",
                "Version": "$Latest"
            },
            "MinSize": 1,
            "MaxSize": 5,
            "DesiredCapacity": 1,
            ...
            "Tags": [
                {
                    "ResourceType": "auto-scaling-group",
                    "ResourceId": "my-asg",
                    "PropagateAtLaunch": true,
```

```
                    "Value": "production",
                    "Key": "environment"
                }
            ],
            ...
        },

    ... additional groups ...

    ]
}
```

Si lo desea, puede especificar las etiquetas mediante un filtro de `tag:<key>`. Por ejemplo, en el siguiente comando se muestra cómo filtrar los resultados para que aparezcan únicamente los grupos de Auto Scaling con una clave de etiqueta y un par de valores de **`environment=production`**. Este filtro tiene el siguiente formato: `Name=tag:<key>,Values=<value>`, con **`<key>`** y **`<value>`** que representan una clave de etiqueta y un par de valores.

```
aws autoscaling describe-auto-scaling-groups \
  --filters Name=tag:environment,Values=production
```

También puede filtrar los AWS CLI resultados mediante la `--query` opción. El siguiente ejemplo muestra cómo limitar el AWS CLI resultado del comando anterior únicamente al nombre del grupo, el tamaño mínimo, el tamaño máximo y los atributos de capacidad deseados.

```
aws autoscaling describe-auto-scaling-groups \
  --filters Name=tag:environment,Values=production \
  --query "AutoScalingGroups[].{AutoScalingGroupName: AutoScalingGroupName, MinSize:
 MinSize, MaxSize: MaxSize, DesiredCapacity: DesiredCapacity}"
```

A continuación, se muestra un ejemplo de respuesta.

```
[
    {
        "AutoScalingGroupName": "my-asg",
        "MinSize": 0,
        "MaxSize": 10,
        "DesiredCapacity": 1
    },

    ... additional groups ...
```

```
]
```

Para obtener más información sobre el filtrado, consulte Filtrar los AWS CLI resultados en la Guía del AWS Command Line Interface usuario.

Ejemplo: describir grupos de Auto Scaling con etiquetas que coincidan con la clave de etiqueta especificada

En el siguiente comando se muestra cómo filtrar los resultados para que aparezcan únicamente los grupos de Auto Scaling con la etiqueta **`environment`**, independientemente del valor de esta.

```
aws autoscaling describe-auto-scaling-groups \
  --filters Name=tag-key,Values=environment
```

Ejemplo: describir grupos de Auto Scaling con etiquetas que coincidan con el conjunto de claves de etiqueta especificadas

En el siguiente comando se muestra cómo filtrar los resultados para que aparezcan únicamente los grupos de Auto Scaling con etiquetas para **`environment`** y **`project`**, independientemente de los valores de estas.

```
aws autoscaling describe-auto-scaling-groups \
  --filters Name=tag-key,Values=environment Name=tag-key,Values=project
```

Ejemplo: describir grupos de Auto Scaling con etiquetas que coincidan con al menos una de las claves de etiqueta especificadas

En el siguiente comando se muestra cómo filtrar los resultados para que aparezcan únicamente los grupos de Auto Scaling con etiquetas para **`environment`** o **`project`**, independientemente de los valores de estas.

```
aws autoscaling describe-auto-scaling-groups \
  --filters Name=tag-key,Values=environment,project
```

Ejemplo: describir grupos de Auto Scaling con el valor de etiqueta especificado

En el siguiente comando se muestra cómo filtrar los resultados para que aparezcan únicamente los grupos de Auto Scaling con un valor de etiqueta de **`production`**, independientemente de la clave de etiqueta.

```
aws autoscaling describe-auto-scaling-groups \
  --filters Name=tag-value,Values=production
```

Ejemplo: describir grupos de Auto Scaling con el conjunto de valores de etiqueta especificados

En el siguiente comando se muestra cómo filtrar los resultados para que aparezcan únicamente los grupos de Auto Scaling con los valores de etiqueta **`production`** y **`development`**, independientemente de la clave de etiqueta.

```
aws autoscaling describe-auto-scaling-groups \
  --filters Name=tag-value,Values=production Name=tag-value,Values=development
```

Ejemplo: describir grupos de Auto Scaling con etiquetas que coincidan con al menos uno de los valores de etiqueta especificados

En el siguiente comando se muestra cómo filtrar los resultados para que aparezcan únicamente los grupos de Auto Scaling con un valor de etiqueta de **`production`** o **`development`**, independientemente de la clave de etiqueta.

```
aws autoscaling describe-auto-scaling-groups \
  --filters Name=tag-value,Values=production,development
```

Ejemplo: describir grupos de Auto Scaling con etiquetas que coincidan con varias claves de etiqueta especificadas

También puede combinar filtros para crear lógicas AND y OR personalizadas y realizar un filtrado más complejo.

En el siguiente comando se muestra cómo filtrar los resultados para que aparezcan únicamente los grupos de Auto Scaling con un conjunto de etiquetas específico. Una clave de etiqueta es **`environment`** AND el valor de la etiqueta es (**`production`** OR **`development`**) AND la otra clave de etiqueta es **`costcenter`** AND el valor de la etiqueta es **`cc123`**.

```
aws autoscaling describe-auto-scaling-groups \
  --filters Name=tag:environment,Values=production,development
 Name=tag:costcenter,Values=cc123
```

Políticas de mantenimiento de instancias

Puede configurar una política de mantenimiento de instancias para que su grupo de escalado automático cumpla con requisitos de capacidad específicos durante los eventos que provocan el reemplazo de las instancias, como la actualización de instancias o el proceso de comprobación de estado.

Por ejemplo, suponga que tiene un grupo de escalado automático con una cantidad reducida de instancias. Desea evitar las posibles interrupciones que pueden provocar la finalización y, a continuación, la sustitución de una instancia cuando las comprobaciones de estado indiquen que la instancia está dañada. Con una política de mantenimiento de instancias, puede asegurarse de que Amazon EC2 Auto Scaling lance primero una nueva instancia y, después, espere a que esté completamente lista antes de finalizar la instancia en mal estado.

Una política de mantenimiento de instancias también lo ayuda a minimizar cualquier posible interrupción en los casos en que se sustituyan varias instancias al mismo tiempo. Usted establece los parámetros de porcentaje de buen estado mínimo y máximo de la política, y su grupo de escalado automático solo puede aumentar y disminuir la capacidad dentro de ese rango mínimo-máximo al reemplazar instancias. Un rango mayor aumenta la cantidad de instancias que se pueden reemplazar al mismo tiempo.

Contenido

- Política de mantenimiento de instancias para el grupo de escalado automático
- Establecimiento de una política de mantenimiento de instancias en el grupo de escalado automático

Política de mantenimiento de instancias para el grupo de escalado automático

Este tema brinda una descripción general de las opciones disponibles y describe qué debe tener en cuenta al crear una política de mantenimiento de instancias.

Contenido

- Descripción general de
- Conceptos clave
- Preparación de las instancias

- Periodo de gracia de la comprobación de estado
- Escalada de su grupo de escalado automático
- Ejemplos de escenarios de

Descripción general de

Cuando crea una política de mantenimiento de instancias para su grupo de escalado automático, la política afecta a los eventos de Amazon EC2 Auto Scaling que provocan la sustitución de las instancias. Esto da como resultado comportamientos de reemplazo más uniformes dentro del mismo grupo de escalado automático. También le permite optimizar la disponibilidad o el costo de su grupo en función de sus necesidades.

En la consola, están disponibles las siguientes opciones de configuración:

- Lance antes de terminar: primero se debe aprovisionar una nueva instancia antes de poder cancelar una instancia existente. Este abordaje es una buena opción para las aplicaciones que prefieren la disponibilidad en lugar del ahorro de costos.
- Finalice y lance: las instancias nuevas se aprovisionan al mismo tiempo que se terminan las instancias existentes. Este abordaje es una buena opción para las aplicaciones que prefieren el ahorro de costos en lugar de la disponibilidad. También es una buena opción para las aplicaciones que no deberían lanzar una capacidad superior a la disponible actualmente, incluso al reemplazar instancias.
- Política personalizada: esta opción permite configurar un rango mínimo y máximo personalizado en la política para la cantidad de capacidad que quiere que esté disponible al reemplazar las instancias. Este enfoque puede ayudarlo a lograr el equilibrio adecuado entre costo y disponibilidad.

El valor predeterminado para un grupo de escalado automático es no tener una política de mantenimiento de instancias, lo que hace que responda a los eventos de mantenimiento de instancias con los comportamientos predeterminados. Los comportamientos predeterminados se describen en la tabla siguiente.

Comportamientos predeterminados de los eventos de mantenimiento de instancias

Event	Description (Descripción)	Comportamiento predeterminado
Error en las comprobaciones de estado	Se produce automáticamente cuando las instancias no superan las comprobaciones de estado. Amazon EC2 Auto Scaling reemplaza las instancias que no superan las comprobaciones de estado. Para conocer las causas de los errores en las comprobaciones de estado, consulte Comprobaciones de estado para instancias en un grupo de escalado automático.	Finalizar y lanzar.
Actualización de instancias	Ocurre cuando inicia una actualización de instancias. Según su configuración, la actualización de instancias reemplaza las instancias de una en una, varias a la vez o todas a la vez. Para obtener más información, consulte Uso de una actualización de instancias para actualizar instancias en un grupo de escalado automático.	Finalizar y lanzar.
Duración máxima de la instancia	Se produce automáticamente cuando las instancias alcanzan la vida útil máxima que especificó para su grupo de escalado automático. Amazon EC2 Auto Scaling	Finalizar y lanzar.

Event	Description (Descripción)	Comportamiento predeterminado
	reemplaza las instancias que alcanzan su máxima vida útil. Para obtener más información, consulte Reemplazo de instancias de Auto Scaling en función de la duración máxima de la instancia.	

Event	Description (Descripción)	Comportamiento predeterminado
Reequilibrio	Se produce automáticamente si hay cambios subyacentes que provocan un desequilibrio en el grupo. Amazon EC2 Auto Scaling reequilibra el grupo en las siguientes situaciones: • Una zona de disponibilidad que anteriormente tenía capacidad insuficiente se recupera o usted agrega o quita una zona de disponibilidad del grupo. Cuando esto sucede, su grupo de escalado automático intenta equilibrarse de manera uniforme en todas las zonas de disponibilidad. Para obtener más información, consulte Actividades de reequilibrio. • Activa el reequilibrio de la capacidad en su grupo de escalado automático e intenta lanzar nuevas instancias de spot antes de que las existentes se interrumpan a medida que cambia la disponibilidad de las instancias de spot. Para obtener más información, consulte Reequilibrio de la capacidad en Auto	Lanzar antes de finalizar. Amazon EC2 Auto Scaling puede superar los límites de tamaño de su grupo hasta en un 10 por ciento de su capacidad máxima. Sin embargo, si utiliza el reequilibrio de la capacidad, solo puede superar estos límites hasta un 10 por ciento de la capacidad deseada.

Event	Description (Descripción)	Comportamiento predeterminado
	Scaling para reemplazar las instancias de spot en riesgo. • Actualiza el grupo de escalado automático y este reemplaza gradualmente las instancias para que coincidan con las nuevas opciones de compra que usted eligió al actualizar una política de instancias mixtas. Para obtener más información, consulte Update an Auto Scaling group.	

Amazon EC2 Auto Scaling seguirá finalizando y lanzando de forma predeterminada en las siguientes situaciones. Por lo tanto, cuando se produce una de estas situaciones, la capacidad de su grupo puede ser menor al umbral inferior de la política de mantenimiento de instancias.

- Cuando una instancia finaliza inesperadamente, por ejemplo, debido a una acción humana. Amazon EC2 Auto Scaling reemplaza inmediatamente las instancias que ya no se ejecutan. Para obtener más información, consulte Comprobaciones de estado de Amazon EC2.
- Cuando Amazon EC2 reinicia, detiene o retira una instancia como parte de un evento programado antes de que Amazon EC2 Auto Scaling pueda lanzar la instancia de reemplazo. Para obtener más información sobre estos eventos, consulte Eventos programados para las instancias en la Guía del usuario de Amazon EC2.
- Cuando el Amazon EC2 Spot Service inicia una interrupción de una instancia de spot y, a continuación, se finaliza forzosamente una instancia de spot.

Con las instancias de spot, si habilitó el reequilibrio de la capacidad en su grupo de escalado automático, es posible que la instancia ya tenga una instancia pendiente de un grupo de spot diferente que lanzamos antes de iniciar la interrupción de spot. Para obtener más información acerca

de cómo funciona el reequilibrio de la capacidad, consulte Reequilibrio de la capacidad en Auto Scaling para reemplazar las instancias de spot en riesgo.

Sin embargo, dado que no se garantiza que las instancias de spot permanezcan disponibles y puedan finalizarse con un aviso de interrupción de dos minutos, se puede superar el límite inferior de su política de mantenimiento de instancias si las instancias se interrumpen antes del lanzamiento de las nuevas instancias.

Conceptos clave

Antes de empezar, familiarícese con los siguientes conceptos y términos centrales:

Capacidad deseada

: La capacidad deseada es la capacidad del grupo de escalado automático al momento de su creación. También es la capacidad que el grupo intenta mantener cuando no hay condiciones de escalado asociadas al grupo.

Política de mantenimiento de instancias

: Una política de mantenimiento de instancias controla si una instancia se aprovisiona primero antes de que finalice una instancia existente en eventos de mantenimiento de instancias. También determina qué tan por debajo y por encima de la capacidad deseada podría llegar su grupo de escalado automático para reemplazar varias instancias al mismo tiempo.

Porcentaje máximo en buen estado

: El porcentaje máximo en buen estado es el porcentaje de la capacidad deseada que su grupo de escalado automático puede aumentar al reemplazar instancias. Representa el porcentaje máximo del grupo que puede estar en servicio y en buen estado, o pendiente, para soportar su carga de trabajo. En la consola, puede establecer el porcentaje máximo en buen estado si utiliza la opción Lanzar antes de finalizar o la opción Política personalizada. Los valores válidos son 100–200 por ciento.

Porcentaje de buen estado mínimo

: El porcentaje de buen estado mínimo es el porcentaje de la capacidad deseada que se quiere mantener en servicio, en buen estado y lista para usarse para soportar la carga de trabajo al reemplazar las instancias. Se considera que una instancia está en buen estado y lista para usarse después de que completa correctamente su primera comprobación de estado y de haber transcurrido el tiempo de preparación especificado. En la consola, puede establecer el porcentaje

de buen estado mínimo si utiliza la opción Finalizar y lanzar o la opción Política personalizada. Los valores válidos son 0-100 por ciento.

> **Note**
>
> Para reemplazar las instancias con mayor rapidez, puede especificar un porcentaje de buen estado mínimo. Sin embargo, si no hay suficientes instancias en buen estado en ejecución, se puede reducir la disponibilidad. Recomendamos seleccionar un valor razonable para mantener la disponibilidad en situaciones en las que se sustituyan varias instancias.

Preparación de las instancias

Si sus instancias necesitan tiempo para inicializarse después de entrar en el estado `InService`, habilite la preparación de instancias predeterminada para su grupo de escalado automático. Con la preparación de instancias predeterminada, puede evitar que las instancias se incluyan en el porcentaje de buen estado mínimo antes de que estén listas. Esto garantiza que Amazon EC2 Auto Scaling considere cuánto tiempo se tarda en disponer de suficiente capacidad para soportar la carga de trabajo antes de finalizar las instancias existentes.

Como ventaja adicional, puedes mejorar las CloudWatch métricas de Amazon utilizadas para el escalado dinámico al habilitar el calentamiento de instancias predeterminado. Si su grupo de Auto Scaling tiene alguna política de escalado, cuando el grupo se amplía, utiliza el mismo período de preparación predeterminado para evitar que las instancias se cuenten para CloudWatch las métricas antes de que hayan terminado de inicializarse.

Para obtener más información, consulte Establecimiento de la preparación predeterminada de instancias para un grupo de escalado automático.

Periodo de gracia de la comprobación de estado

Amazon EC2 Auto Scaling determina si una instancia está en buen estado en función de las comprobaciones de estado que utiliza su grupo de escalado automático. Para obtener más información, consulte Comprobaciones de estado para instancias en un grupo de escalado automático.

Para asegurarse de que estas comprobaciones de estado comiencen lo antes posible, no establezca demasiado alto el periodo de gracia de las comprobaciones de estado del grupo, sino

lo suficientemente alto como para que las comprobaciones de estado de Elastic Load Balancing puedan determinar si hay un objetivo disponible para gestionar las solicitudes. Para obtener más información, consulte Establezca el periodo de gracia de la comprobación de estado para un grupo de escalado automático.

Escalada de su grupo de escalado automático

Una política de mantenimiento de instancias solo se aplica a los eventos de mantenimiento de instancias y no impide que el grupo se escale manual o automáticamente.

Cuando hay políticas de escalado o acciones programadas asociadas a su grupo de escalado automático, pueden ejecutarse en paralelo mientras se producen los eventos de mantenimiento de la instancia. En ese caso, podrían aumentar o disminuir la capacidad deseada del grupo, pero solo dentro de los límites de escalado que usted haya definido. Para obtener más información sobre estos límites, consulte Establecimiento de límites de escalado para el grupo de escalado automático.

Ejemplos de escenarios de

En un escenario típico, la política de mantenimiento de instancias y la capacidad deseada podrían tener un aspecto similar al siguiente:

- Porcentaje de buen estado mínimo = 90 por ciento
- Porcentaje máximo en buen estado = 120 por ciento
- Capacidad deseada = 100

Durante cualquier evento de mantenimiento de instancias, su grupo de escalado automático puede tener entre 90 y 120 instancias. Tras el evento, el grupo vuelve a tener 100 instancias.

Cuando utiliza una política de mantenimiento de instancias con un grupo de escalado automático que tiene un grupo en caliente, los porcentajes mínimo y máximo de buen estado se aplican por separado al grupo de escalado automático y al grupo en caliente.

Por ejemplo, supongamos que esta es su configuración:

- Porcentaje de buen estado mínimo = 90 por ciento
- Porcentaje máximo en buen estado = 120 por ciento
- Capacidad deseada = 100
- Tamaño del grupo en caliente = 10

Si inicia una actualización de instancias para reciclar las instancias del grupo, Amazon EC2 Auto Scaling reemplaza primero las instancias del grupo de escalado automático y, después, las instancias del grupo en caliente. Si bien Amazon EC2 Auto Scaling sigue trabajando en la sustitución de instancias del grupo de escalado automático, el grupo puede tener entre 90 y 120 instancias. Después de terminar con el grupo, Amazon EC2 Auto Scaling puede trabajar en la sustitución de instancias del grupo en caliente. Mientras esto sucede, el grupo en caliente podría tener entre 9 y 12 instancias.

Establecimiento de una política de mantenimiento de instancias en el grupo de escalado automático

Puede crear una política de mantenimiento de instancias cuando cree un grupo de escalado automático. También puede crearla para grupos existentes.

Al establecer una política de mantenimiento de instancias en su grupo de escalado automático, ya no tiene que especificar valores para los parámetros de porcentaje mínimo y máximo de buen estado para la característica de actualización de instancias, a menos que quiera anular la política de mantenimiento de instancias.

En la consola, Amazon EC2 Auto Scaling ofrece opciones para ayudarlo a comenzar.

Contenido

- Establecimiento de una política de mantenimiento de instancias
- Eliminación de una política de mantenimiento de instancias

Establecimiento de una política de mantenimiento de instancias

Para establecer una política de mantenimiento de instancias en un grupo de escalado automático, use uno de los siguientes métodos:

Console

Establecimiento de una política de mantenimiento de instancias en un grupo (consola)

1. Siga las instrucciones de Creación de un grupo de Auto Scaling mediante una plantilla de lanzamiento y complete cada paso del procedimiento, hasta el paso 11.
2. En Configurar el tamaño del grupo y las políticas de escalado, en Capacidad deseada, introduzca la cantidad inicial de instancias que se van a lanzar.

3. En la sección Escalado, en Límites de escalado, si el nuevo valor de Capacidad deseada es superior a Capacidad deseada mínima y a Capacidad deseada máxima, Capacidad deseada máxima se incrementará automáticamente al nuevo valor de capacidad deseada. Puede cambiar estos límites según sea necesario.
4. En Escalado automático, elija si desea crear una política de escalado de seguimiento de destino. También puede crear esta política después de crear su grupo de escalado automático.

   Si elige Política de escalado de seguimiento de destino, siga las instrucciones en Creación de una política de escalado de seguimiento de destino para crear la política.
5. En la sección Política de mantenimiento de instancias, elija una de las opciones disponibles:

   - Lance antes de terminar: primero se debe aprovisionar una nueva instancia antes de poder cancelar una instancia existente. Esta es una buena opción para las aplicaciones que prefieren la disponibilidad en lugar del ahorro de costos.
   - Finalice y lance: las instancias nuevas se aprovisionan al mismo tiempo que se terminan las instancias existentes. Esta es una buena opción para las aplicaciones que favorecen el ahorro de costos por encima de la disponibilidad. También es una buena opción para las aplicaciones que no deberían lanzar una capacidad superior a la disponible actualmente.
   - Política personalizada: esta opción permite configurar un rango mínimo y máximo personalizado en la política para la cantidad de capacidad que quiere que esté disponible al reemplazar las instancias. Esto puede ayudarlo a lograr el equilibrio adecuado entre costo y disponibilidad.
6. En Defina un porcentaje de buen estado, introduzca valores para uno o ambos de los siguientes campos. Los campos habilitados varían en función de la opción que haya elegido en el paso anterior.

   - Mínimo: establece el porcentaje de buen estado mínimo necesario para proceder con el reemplazo de instancias.
   - Máximo: establece el porcentaje máximo en buen estado posible cuando se reemplazan instancias.
7. Amplíe la sección Ver la capacidad durante las sustituciones en función de la capacidad deseada para confirmar cómo se aplican los valores mínimo y máximo a su grupo. Los valores exactos utilizados dependen del valor de la capacidad deseada, que cambiará si el grupo escala.

8. Continúe con los pasos en Creación de un grupo de Auto Scaling mediante una plantilla de lanzamiento.

AWS CLI

Establecimiento de una política de mantenimiento de instancias en un grupo nuevo (AWS CLI)

Añada la `--instance-maintenance-policy` opción al comando. create-auto-scaling-group El siguiente ejemplo establece una política de mantenimiento de instancias en un nuevo grupo de escalado automático denominado *my-asg*.

```
aws autoscaling create-auto-scaling-group \
  --launch-template LaunchTemplateName=my-launch-template,Version='1' \
  --auto-scaling-group-name my-asg \
  --min-size 1 \
  --max-size 10 \
  --desired-capacity 5 \
  --default-instance-warmup 20 \
  --instance-maintenance-policy '{
      "MinHealthyPercentage": 90,
      "MaxHealthyPercentage": 120
    }' \
  --vpc-zone-identifier "subnet-5e6example,subnet-613example,subnet-c93example"
```

Console

Establecimiento de una política de mantenimiento de instancias en un grupo de existente (consola)

1. Abra la consola Amazon EC2 en https://console.aws.amazon.com/ec2/y seleccione Auto Scaling Groups en el panel de navegación.
2. En la barra de navegación de la parte superior de la pantalla, elija la misma Región de AWS que utilizó cuando creó el grupo de escalado automático.
3. Seleccione la casilla situada junto al grupo de escalado automático.

   Se abre un panel dividido en la parte inferior de la página.
4. En la pestaña Detalles, elija Política de mantenimiento de instancias, Editar.
5. Para establecer una política de mantenimiento de instancias en el grupo, elija una de las opciones disponibles:

- Lance antes de terminar: primero se debe aprovisionar una nueva instancia antes de poder cancelar una instancia existente. Esta es una buena opción para las aplicaciones que prefieren la disponibilidad en lugar del ahorro de costos.
- Finalice y lance: las instancias nuevas se aprovisionan al mismo tiempo que se terminan las instancias existentes. Esta es una buena opción para las aplicaciones que favorecen el ahorro de costos por encima de la disponibilidad. También es una buena opción para las aplicaciones que no deberían lanzar una capacidad superior a la disponible actualmente.
- Política personalizada: esta opción permite configurar un rango mínimo y máximo personalizado en la política para la cantidad de capacidad que quiere que esté disponible al reemplazar las instancias. Esto puede ayudarlo a lograr el equilibrio adecuado entre costo y disponibilidad.

6. En Defina un porcentaje de buen estado, introduzca valores para uno o ambos de los siguientes campos. Los campos habilitados varían en función de la opción que haya elegido en el paso anterior.

   - Mínimo: establece el porcentaje de buen estado mínimo necesario para proceder con el reemplazo de instancias.
   - Máximo: establece el porcentaje máximo en buen estado posible cuando se reemplazan instancias.

7. Amplíe la sección Ver la capacidad durante las sustituciones en función de la capacidad deseada para confirmar cómo se aplican los valores mínimo y máximo a su grupo. Los valores exactos utilizados dependen del valor de la capacidad deseada, que cambiará si el grupo escala.
8. Elija Actualizar.

AWS CLI

Establecimiento de una política de mantenimiento de instancias en un grupo existente (AWS CLI)

Añada la `--instance-maintenance-policy` opción al update-auto-scaling-groupcomando. El siguiente ejemplo establece una política de mantenimiento de instancias en el grupo de escalado automático especificado.

```
aws autoscaling update-auto-scaling-group --auto-scaling-group-name my-asg \
  --instance-maintenance-policy '{
      "MinHealthyPercentage": 90,
```

```
    "MaxHealthyPercentage": 120
  }'
```

Eliminación de una política de mantenimiento de instancias

Si desea dejar de utilizar una política de mantenimiento de instancias con su grupo de escalado automático, puede eliminarla.

Console

Eliminación de una política de mantenimiento de instancias (consola)

1. Abra la consola Amazon EC2 en https://console.aws.amazon.com/ec2/y seleccione Auto Scaling Groups en el panel de navegación.
2. En la barra de navegación de la parte superior de la pantalla, elija la misma Región de AWS que utilizó cuando creó el grupo de escalado automático.
3. Seleccione la casilla situada junto al grupo de escalado automático.

   Se abre un panel dividido en la parte inferior de la página.
4. En la pestaña Detalles, elija Política de mantenimiento de instancias, Editar.
5. Seleccione Sin política de mantenimiento de instancias.
6. Elija Actualizar.

AWS CLI

Eliminación de una política de mantenimiento de instancias (AWS CLI)

Añada la `--instance-maintenance-policy` opción al update-auto-scaling-groupcomando. El siguiente ejemplo elimina la política de mantenimiento de instancias del grupo de escalado automático especificado.

```
aws autoscaling update-auto-scaling-group --auto-scaling-group-name my-asg \
  --instance-maintenance-policy '{
      "MinHealthyPercentage": -1,
      "MaxHealthyPercentage": -1
    }'
```

Enlaces de ciclo de vida de Amazon EC2 Auto Scaling

Amazon EC2 Auto Scaling ofrece la posibilidad de agregar enlaces de ciclo de vida a los grupos de Auto Scaling. Estos enlaces le permiten crear soluciones que tengan en cuenta los eventos del ciclo de vida de la instancia de Auto Scaling y que, a continuación, realicen una acción personalizada en las instancias cuando se produzca el evento del ciclo de vida correspondiente. Un enlace de ciclo de vida proporciona una cantidad de tiempo especificada (una hora de forma predeterminada) para esperar a que se complete la acción antes de que la instancia pase al siguiente estado.

Como ejemplo del uso de enlaces de ciclo de vida con instancias de Auto Scaling:

- Cuando ocurre un evento de escalado horizontal, la instancia que se acaba de lanzar completa la secuencia de arranque y pasa a un estado de espera. Mientras la instancia se encuentra en un estado de espera, ejecuta un script para descargar e instalar los paquetes de software necesarios para la aplicación, asegurándose de que la instancia esté completamente lista antes de que comience a recibir tráfico. Cuando el script ha terminado de instalar el software, envía el comando complete-lifecycle-action para continuar.
- Cuando se produce un evento de escalado, un enlace del ciclo de vida detiene la instancia antes de que finalice y te envía una notificación a través de Amazon. EventBridge Mientras la instancia esté en estado de espera, puedes invocar una AWS Lambda función o conectarte a la instancia para descargar los registros u otros datos antes de que la instancia finalice por completo.

Un uso popular de los enlaces de ciclo de vida es controlar cuándo se registran instancias en Elastic Load Balancing. Si agrega un enlace de ciclo de vida de lanzamiento al grupo de Auto Scaling, puede asegurarse de que los scripts de arranque se hayan completado correctamente y que las aplicaciones de las instancias estén listas para aceptar tráfico antes de que se registren en el balanceador de carga al final del enlace de ciclo de vida.

Contenido

- Disponibilidad de los enlaces de ciclo de vida
- Consideraciones y limitaciones de enlaces de ciclo de vida
- Recursos relacionados
- Cómo funcionan los enlaces de ciclo de vida en los grupos de escalado automático
- Preparación para agregar un enlace de ciclo de vida a un grupo de Auto Scaling
- Controle la retención de instancias con políticas de ciclo de vida de las instancias

- Recuperar el estado de ciclo de vida de destino a través de los metadatos de instancia
- Incorporación de los enlaces de ciclo de vida en su grupo de escalado automático
- Cómo completar una acción del ciclo de vida en un grupo de escalado automático
- Tutorial: Uso del script de datos y los metadatos de la instancia para recuperar el estado del ciclo de vida
- Tutorial: Configuración de un enlace de ciclo de vida que invoca una función Lambda

Disponibilidad de los enlaces de ciclo de vida

En la tabla siguiente se enumeran los enlaces de ciclo de vida disponibles para varias situaciones.

Event	Inicio o terminación de la instancia[1]	Duración máxima de instancia: instancias de reemplazo	Actualización de instancia: instancias de reemplazo	Reequilibrio de capacidad: instancias de reemplazo	Grupos de calentamiento: instancias que entran y salen del grupo de calentamiento
Lanzamiento de la instancia	✓	✓	✓	✓	✓
Terminación de la instancia	✓	✓	✓	✓	✓

[1] Se aplica a todos los lanzamientos y terminaciones, ya sea que se inicien de forma automática o manual, como cuando se llama a las operaciones `SetDesiredCapacity` o `TerminateInstanceInAutoScalingGroup`. No se aplica al adjuntar o desconectar instancias, al poner o sacar instancias del modo de espera, o al eliminar el grupo con la opción para forzar la eliminación.

Consideraciones y limitaciones de enlaces de ciclo de vida

Cuando trabaje con enlaces de ciclo de vida, tenga en cuenta las siguientes notas y limitaciones:

- Amazon EC2 Auto Scaling proporciona su propio ciclo de vida para ayudar con la administración de grupos de Auto Scaling. Este ciclo de vida difiere del de otras instancias EC2. Para obtener más información, consulte Ciclo de vida de instancias de Amazon EC2 Auto Scaling. Las instancias de un grupo de calentamiento también tienen su propio ciclo de vida, como se describe en Transiciones de estado del ciclo de vida para las instancias de un grupo de calentamiento.
- Si configura un grupo activo para su grupo de Auto Scaling, Amazon EC2 Auto Scaling detiene o hiberna cada instancia que entra en el grupo activo. La acción de detener o hibernar no espera a que los datos del usuario terminen de ejecutarse. Para retener una instancia hasta que finalice el script de arranque, usa un enlace de ciclo de vida para la acción del ciclo de `EC2_INSTANCE_LAUNCHING` vida. Para obtener más información, consulte Uso de enlaces de ciclo de vida con un grupo de calentamiento en un grupo de escalado automático.
- De forma predeterminada, los ganchos del ciclo de vida de terminación funcionan en la medida de lo posible. Si se agota el tiempo de espera de un ciclo de vida de terminación o se abandona, Amazon EC2 Auto Scaling procederá a la terminación inmediata de la instancia. Puede combinar los ganchos del ciclo de vida de terminación con una política del ciclo de vida de las instancias para la retención de las instancias. Para obtener más información, consulte Controle la retención de instancias con políticas de ciclo de vida de las instancias.
- Puede utilizar enlaces de ciclo de vida con instancias de spot, pero un enlace de ciclo de vida no impide que se termine una instancia si esa capacidad ya no está disponible, lo que puede suceder en cualquier momento con un aviso de interrupción de dos minutos. Para obtener más información, consulte Interrupciones de instancias de spot en la Guía del usuario de Amazon EC2. Sin embargo, puede habilitar el reequilibrio de la capacidad para reemplazar de forma proactiva las instancias de spot que han recibido una recomendación de reequilibrio del servicio de spot de Amazon EC2, una señal que se envía cuando una instancia de spot tiene un riesgo elevado de interrupción. Para obtener más información, consulte Reequilibrio de la capacidad en Auto Scaling para reemplazar las instancias de spot en riesgo.
- Las instancias pueden permanecer en un estado de espera durante un periodo de tiempo limitado. El tiempo de espera predeterminado para un enlace de ciclo de vida es de una hora (tiempo de espera de latido). También hay un tiempo de espera global que especifica la cantidad máxima de tiempo que se puede mantener una instancia en estado de espera. El tiempo de espera global es de 48 horas, o 100 veces el tiempo de espera de latido, el que sea más corto.

- El resultado del enlace de ciclo de vida puede ser “abandonar” o “continuar”. Si una instancia se está lanzando, “continuar” indica que las acciones se han realizado correctamente y que Amazon EC2 Auto Scaling puede poner la instancia en servicio. Por el contrario, “abandonar” indica que las acciones personalizadas no se realizaron correctamente y que podemos terminar la instancia y reemplazarla. Si una instancia está terminando, tanto “abandonar” como “continuar” permiten terminar la instancia. Sin embargo, abandonar detiene todas las acciones restantes, como otros enlaces de ciclo de vida, mientras que continuar permite que todos los demás enlaces de ciclo de vida se completen.
- Amazon EC2 Auto Scaling limita la velocidad a la que permite el lanzamiento de instancias si los enlaces de ciclo de vida fallan de forma constante, de modo que asegúrese de probar y corregir cualquier error permanente en las acciones de ciclo de vida.
- La creación y la actualización de los ganchos del ciclo de vida mediante el AWS CLI CloudFormation, o un SDK ofrecen opciones que no están disponibles al crear un enlace del ciclo de vida desde el Consola de administración de AWS. Por ejemplo, el campo para especificar el ARN de un tema de SNS o una cola de SQS no aparece en la consola porque Amazon EC2 Auto Scaling ya envía eventos a Amazon. EventBridge Estos eventos se pueden filtrar y redirigir a AWS servicios como Lambda, Amazon SNS y Amazon SQS, según sea necesario.
- Puede añadir varios enlaces de ciclo de vida a un grupo de Auto Scaling mientras lo crea. Para ello, llame a la CreateAutoScalingGroup API mediante el SDK AWS CLI o el CloudFormation SDK. No obstante, cada enlace debe tener el mismo destino de notificación y rol de IAM, si se especifica. Para crear ganchos de ciclo de vida con diferentes objetivos de notificación y diferentes roles, crea los ganchos de ciclo de vida uno a uno en distintas llamadas a la PutLifecycleHook API.
- Si agrega un enlace de ciclo de vida para el lanzamiento de instancias, el periodo de gracia de la comprobación de estado comienza en cuanto la instancia alcance el estado `InService`. Para obtener más información, consulte Establezca el periodo de gracia de la comprobación de estado para un grupo de escalado automático.

Consideraciones de escalado

- Las políticas de escalado dinámico se amplían y reducen en respuesta a los datos CloudWatch métricos, como la CPU y la red I/O, que se agregan en varias instancias. Durante un escalado horizontal, Amazon EC2 Auto Scaling no tiene en cuenta inmediatamente una nueva instancia en las métricas agrupadas de la instancia del grupo de escalado automático. Espera a que la instancia alcance el estado `InService` y finalice el calentamiento de la instancia. Para obtener

más información, consulte Consideraciones sobre el rendimiento de escalado en el tema de calentamiento predeterminado de instancias.

- Cuando se reduce horizontalmente, es posible que las métricas de instancias agregadas no reflejen al instante la eliminación de una instancia de terminación. La instancia que termina deja de contar con respecto a las métricas agrupadas de la instancia del grupo poco después Amazon EC2 Auto Scaling comience el flujo de trabajo de terminación.
- En la mayoría de los casos, cuando se invocan enlaces de ciclo de vida, las actividades de escalado debidas a políticas de escalado sencillo se ponen en pausa hasta que se hayan completado las acciones de ciclo de vida y el periodo de recuperación haya vencido. Si se establece un intervalo largo para el periodo de recuperación, el escalado tardará más tiempo en reanudarse. Para obtener más información, consulte Los enlaces de ciclo de vida pueden provocar retrasos adicionales en el tema sobre recuperación. En general, le recomendamos que no utilice políticas de escalado simple si en su lugar puede usar políticas de escalado por pasos o de escalado de seguimiento de destino.

Recursos relacionados

Para ver un vídeo de presentación, consulte AWS re:Invent 2018: Capacity Management Made Easy with Amazon EC2 Auto Scaling on. YouTube

Te proporcionamos algunos fragmentos de plantillas JSON y YAML que puedes usar para entender cómo declarar los ganchos del ciclo de vida en tus plantillas de pila. CloudFormation Para obtener más información, consulta la LifecycleHook referencia AWS::AutoScaling:: en la Guía del AWS CloudFormation usuario.

También puedes visitar nuestro GitHub repositorio para descargar plantillas de ejemplo y scripts de datos de usuario para vincular el ciclo de vida.

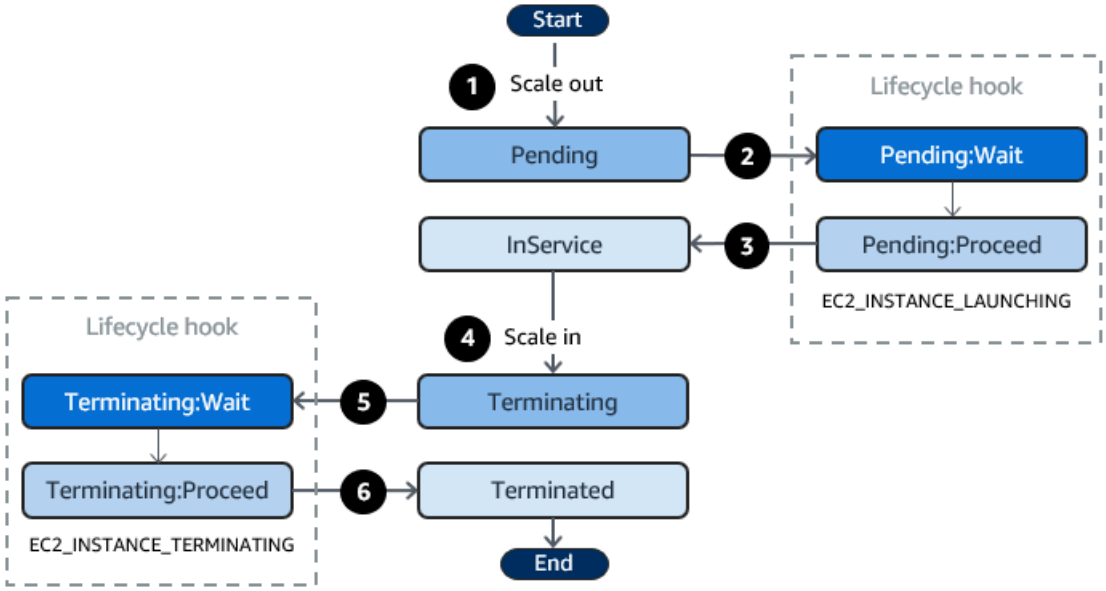
Para ver ejemplos del uso de enlaces de ciclo de vida, consulte las siguientes entradas del blog.

- Building a Backup System for Scaled Instances using Lambda and Amazon EC2 Run Command
- Ejecute el código antes de finalizar una instancia de escalado automático de EC2.

Cómo funcionan los enlaces de ciclo de vida en los grupos de escalado automático

Una instancia de Amazon EC2 pasa por diferentes estados desde el momento en que la lanza hasta que termina. Puede crear acciones personalizadas para su grupo de escalado automático a fin de que actúen cuando una instancia pasa a un estado de espera a causa de un enlace de ciclo de vida.

En la siguiente ilustración se muestran las transiciones entre los estados de la instancia de escalado automático cuando utiliza los enlaces de ciclo de vida para escalar y reducir horizontalmente.



Como se muestra en el diagrama anterior:

1. El grupo de Auto Scaling responde a un evento de escalado horizontal y comienza a iniciar una instancia.
2. El enlace de ciclo de vida pone la instancia en estado de espera (`Pending:Wait`) y luego ejecuta una acción personalizada.

   La instancia permanece en estado de espera hasta que se completa la acción de ciclo de vida o finaliza el periodo de tiempo de espera. De forma predeterminada, la instancia permanece en estado de espera durante una hora y, a continuación, el grupo de Auto Scaling continúa con el proceso de inicio (`Pending:Proceed`). Si necesita más tiempo, puede reiniciar el periodo de tiempo de espera registrando un latido. Si se completa la acción de ciclo de vida cuando la acción personalizada se ha realizado y el periodo de tiempo de espera no ha vencido aún, el periodo finaliza y el grupo de Auto Scaling continúa con el proceso de lanzamiento.

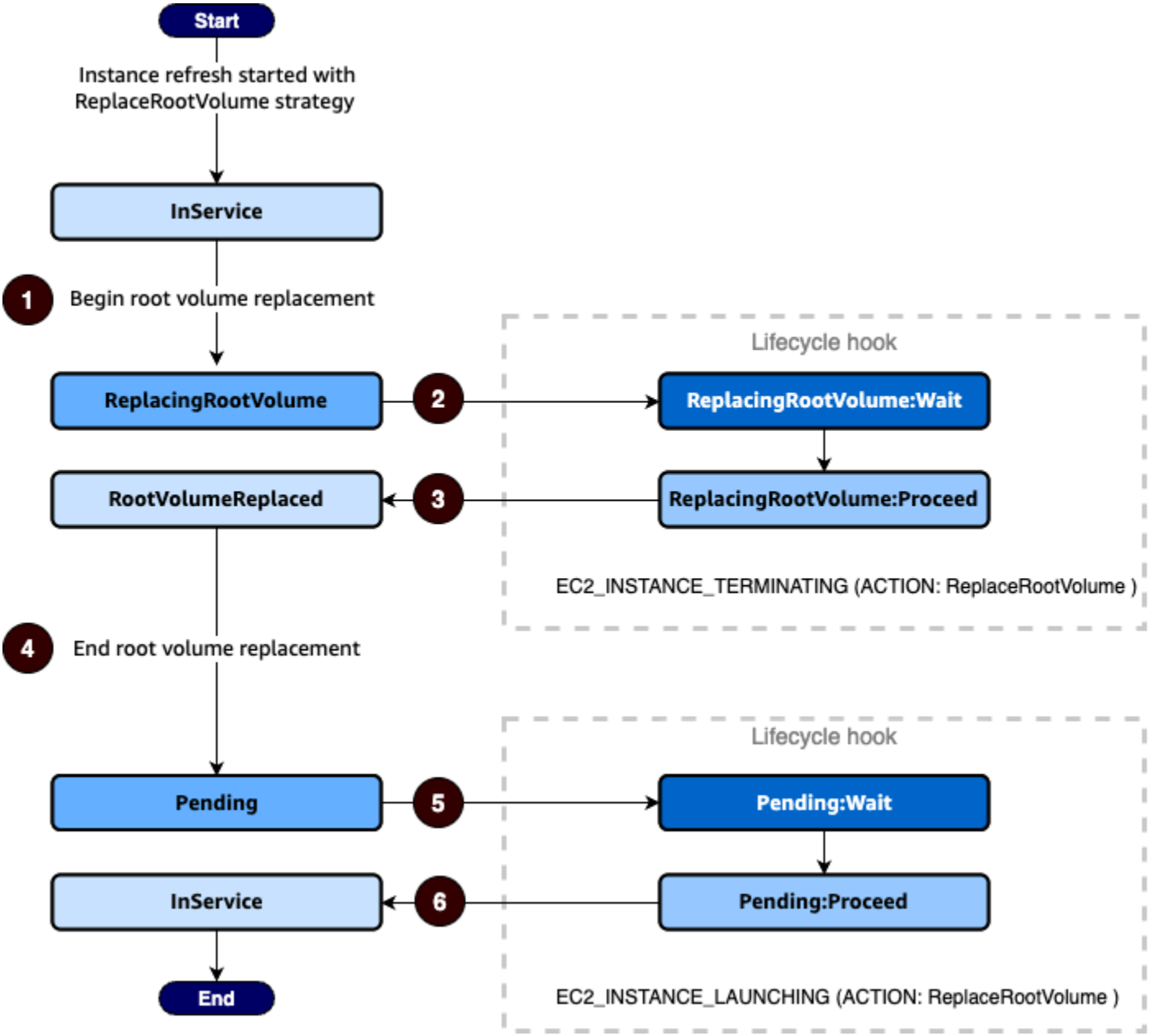
3. La instancia pasa al estado `InService` y comienza el periodo de gracia de la comprobación de estado. Sin embargo, antes de que la instancia alcance el estado `InService`, si el grupo de Auto Scaling está asociado a un balanceador de carga de Elastic Load Balancing, la instancia se registra en el balanceador de carga, y este comienza a comprobar su estado. Una vez que termina el periodo de gracia de la comprobación de estado, Amazon EC2 Auto Scaling comienza a comprobar el estado de la instancia.
4. El grupo de Auto Scaling responde a un evento de reducción horizontal y comienza a terminar una instancia. Si el grupo de Auto Scaling se usa con Elastic Load Balancing, primero el registro de la instancia que terminará se anula del balanceador de carga. Si Connection Draining está habilitado para el balanceador de carga, la instancia deja de aceptar nuevas conexiones y espera a que las conexiones existentes se agoten antes de completar el proceso de anulación del registro.
5. El enlace de ciclo de vida pone la instancia en estado de espera (`Terminating:Wait`) y luego realiza una acción personalizada.

   La instancia permanece en estado de espera hasta que se completa la acción del ciclo de vida o hasta que finaliza el tiempo de espera (que, de forma predeterminada, es de una hora). Después de completar el enlace de ciclo de vida o de que el tiempo de espera expire, la instancia pasa al siguiente estado (`Terminating:Proceed`).
6. La instancia se termina.

> **Important**
>
> Las instancias de un grupo de calentamiento también tienen su propio ciclo de vida con los estados de espera correspondientes, como se describe en Transiciones de estado del ciclo de vida para las instancias de un grupo de calentamiento.

Transiciones entre los estados del ciclo de vida de las instancias que están siendo reemplazadas por

En el siguiente diagrama, se muestra la transición entre los estados de las instancias de Auto Scaling cuando se utilizan ganchos de ciclo de vida para reemplazar el volumen raíz:



Como se muestra en el diagrama anterior:

1. El grupo de Auto Scaling responde a la actualización de una instancia y selecciona una instancia para sustituirla por el volumen raíz. La instancia entra en el `ReplacingRootVolume` estado. Si la instancia está registrada con un balanceador de cargas, se cancela su registro en el balanceador de cargas.
2. El enlace de ciclo de vida pone la instancia en estado de espera (`ReplacingRootVolume:Wait`) y luego ejecuta una acción personalizada. La instancia permanece en estado de espera hasta que se completa la acción de ciclo de vida o finaliza el periodo de tiempo de espera. Si completas

la acción del ciclo de vida cuando se haya completado la acción personalizada y el período de espera aún no haya expirado, el período finaliza y el grupo de Auto Scaling continúa con el proceso de reemplazo del volumen raíz.

3. La instancia completa el reemplazo del volumen raíz y entra en el `RootVolumeReplaced` estado.
4. La instancia entra en el `Pending` estado.
5. El enlace de ciclo de vida pone la instancia en estado de espera (`Pending:Wait`) y luego ejecuta una acción personalizada. La instancia permanece en estado de espera hasta que se complete la acción del ciclo de vida o hasta que finalice el período de tiempo de espera. Después de completar el enlace de ciclo de vida o de que el tiempo de espera expire, la instancia pasa al siguiente estado (`Pending:Proceed`).
6. La instancia entra en el `InService` estado. Sin embargo, antes de que la instancia alcance el `InService` estado, si el grupo de Auto Scaling está asociado a un balanceador de cargas de Elastic Load Balancing, la instancia se registra en el balanceador de cargas.

Preparación para agregar un enlace de ciclo de vida a un grupo de Auto Scaling

Antes de agregar un enlace de ciclo de vida al grupo de Auto Scaling, asegúrese de que el script de datos de usuario o el destino de notificación estén configurados correctamente.

- No es necesario configurar un destino de notificación para utilizar un script de datos de usuario con el fin de realizar acciones personalizadas en las instancias mientras se están lanzando. Sin embargo, ya debe haber creado la plantilla de lanzamiento o la configuración de lanzamiento que especifica el script de datos de usuario y debe haberla asociado al grupo de Auto Scaling. Para obtener más información acerca de los scripts de datos de usuario, consulte Ejecutar comandos en la instancia de Linux durante la inicialización en la Guía del usuario de Amazon EC2.
- Para indicar a Amazon EC2 Auto Scaling que la acción del ciclo de vida ha finalizado, debe añadir la llamada a la CompleteLifecycleAction API al script y crear manualmente un rol de IAM con una política que permita a las instancias de Auto Scaling llamar a esta API. La plantilla de lanzamiento o la configuración de lanzamiento deben especificar este rol mediante un perfil de instancias de IAM que se adjunta a las instancias de Amazon EC2 en el momento del lanzamiento. Para obtener más información, consulte Cómo completar una acción del ciclo de vida en un grupo de escalado automático y Rol de IAM para aplicaciones que se ejecuten en instancias de Amazon EC2.
- Para permitir que Lambda envíe una señal a Amazon EC2 Auto Scaling cuando se haya completado la acción del ciclo de vida, debe añadir la llamada a la CompleteLifecycleAction API

al código de la función. También debe haber adjuntado una política de IAM al rol de ejecución de la función que concede permiso a Lambda para completar acciones de ciclo de vida. Para obtener más información, consulte Tutorial: Configuración de un enlace de ciclo de vida que invoca una función Lambda.

- Para utilizar un servicio como Amazon SNS o Amazon SQS para realizar una acción personalizada, ya debe haber creado el tema SNS o la cola SQS y tener preparado su nombre de recurso de Amazon (ARN). También debe haber creado el rol de IAM que otorga a Amazon EC2 Auto Scaling acceso al tema SNS o al destino de SQS y tener preparado su ARN. Para obtener más información, consulte Configuración de un destino de notificación para notificaciones de ciclo de vida.

> **Note**
>
> De forma predeterminada, al añadir un enlace de ciclo de vida en la consola, Amazon EC2 Auto Scaling envía notificaciones de eventos del ciclo de vida a Amazon. EventBridge Se recomienda utilizar un script de datos de usuario EventBridge o utilizar un script de datos de usuario. Para crear un enlace de ciclo de vida que envíe notificaciones directamente a Amazon SNS o Amazon SQS AWS Lambda, utilice el AWS CLI enlace de ciclo de vida o un SDK para añadir el enlace de ciclo de vida. AWS CloudFormation

Configuración de un destino de notificación para notificaciones de ciclo de vida

Puede agregar enlaces de ciclo de vida a un grupo de Auto Scaling para realizar acciones personalizadas cuando una instancia entra en estado de espera. Puede elegir un servicio de destino para llevar a cabo estas acciones en función de su enfoque de desarrollo preferido.

Existen cuatro enfoques diferentes para implementar los objetivos de notificación para los enlaces de ciclo de vida:

- Amazon EventBridge: reciba las notificaciones y realice las acciones que desee.
- Amazon Simple Notification Service (Amazon SNS): cree un tema para publicar notificaciones. Los clientes pueden suscribirse al tema de SNS y recibir mensajes publicados mediante un protocolo compatible.
- Amazon Simple Queue Service (Amazon SQS): intercambie mensajes mediante un modelo de sondeo.
- AWS Lambda: invoque una función de Lambda que ejecute la acción deseada.

Como práctica recomendada, le recomendamos que utilice Amazon SNS o Amazon SQS como objetivo de notificación en EventBridge lugar de Amazon SNS. Las notificaciones enviadas a Amazon SNS y Amazon SQS contienen la misma información que las notificaciones a las que envía Amazon EC2 Auto Scaling. EventBridge Antes EventBridge, la práctica estándar consistía en enviar una notificación a SNS o SQS e integrar otro servicio con SNS o SQS para realizar acciones programáticas. En la actualidad, EventBridge ofrece más opciones a los servicios a los que puede dirigirse y facilita la gestión de los eventos mediante una arquitectura sin servidor.

Recuerde que si tiene un script de datos de usuario en la plantilla de lanzamiento o en la configuración de lanzamiento que configura las instancias cuando se lanzan, no es necesario que reciba notificaciones para llevar a cabo acciones personalizadas en las instancias.

Los siguientes procedimientos abordan cómo configurar el destino de notificaciones.

Contenido

- Envíe las notificaciones a Lambda mediante EventBridge
- Recepción de notificaciones mediante Amazon SNS
- Recepción de notificaciones mediante Amazon SQS
- Dirige las notificaciones a AWS Lambda directamente
- Ejemplo de mensaje de notificación

> **Important**
>
> La EventBridge regla, la función de Lambda, el tema de Amazon SNS y la cola de Amazon SQS que utilice con los enlaces del ciclo de vida deben estar siempre en la misma región en la que creó el grupo de Auto Scaling.

Envíe las notificaciones a Lambda mediante EventBridge

Puedes configurar una EventBridge regla para invocar una función de Lambda cuando una instancia entre en estado de espera. Amazon EC2 Auto Scaling emite una notificación de eventos del ciclo de vida EventBridge sobre la instancia que se está lanzando o finalizando y un token que puede usar para controlar la acción del ciclo de vida. Para ver ejemplos de estos eventos, consulte Referencia de evento de Amazon EC2 Auto Scaling.

> **Note**
>
> Cuando la utiliza Consola de administración de AWS para crear una regla de eventos, la consola añade automáticamente los permisos de IAM necesarios para conceder el EventBridge permiso de llamar a la función de Lambda. Si crea una regla de eventos utilizando la AWS CLI, tiene que otorgar este permiso explícitamente.
> Para obtener información sobre cómo crear reglas de eventos en la EventBridge consola, consulte Cómo crear EventBridge reglas de Amazon que reaccionen a los eventos en la Guía del EventBridge usuario de Amazon.
> - o bien -
> Para obtener una guía introductoria dirigida a los usuarios de la consola, consulte Tutorial: Configuración de un enlace de ciclo de vida que invoca una función Lambda. En este tutorial se muestra cómo crear una función Lambda sencilla que detecte los eventos de lanzamiento y los escriba en un CloudWatch registro de registros.

Para crear una EventBridge regla que invoque una función de Lambda

1. Cree una función Lambda mediante la consola de Lambda y anote el nombre de recurso de Amazon (ARN). Por ejemplo, `arn:aws:lambda:`*`region`*`:123456789012:function:`*`my-function`*. Necesitas el ARN para crear un objetivo. EventBridge Para obtener más información, consulte Introducción a Lambda en la Guía para desarrolladores de AWS Lambda .
2. Para crear una regla que coincida con los eventos de un lanzamiento de instancia, use el siguiente comando put-rule.

   ```
   aws events put-rule --name my-rule --event-pattern file://pattern.json --state
    ENABLED
   ```

   En el siguiente ejemplo se muestra `pattern.json` para una acción del ciclo de vida de inicio de una instancia. Sustituya el texto por el nombre de su grupo de Auto Scaling. *italics*

   ```
   {
     "source": [ "aws.autoscaling" ],
     "detail-type": [ "EC2 Instance-launch Lifecycle Action" ],
     "detail": {
         "AutoScalingGroupName": [ "my-asg" ]
      }
   }
   ```

Si el comando se ejecuta correctamente, EventBridge responde con el ARN de la regla. Anote este ARN. Tendrá que ingresarlo en el paso 4.

Para crear una regla que coincida con otros eventos, modifique el patrón de eventos. Para obtener más información, consulte Se usa EventBridge para gestionar eventos de Auto Scaling.

3. Para especificar la función Lambda que se va a utilizar como destino de la regla, utilice el siguiente comando put-targets.

```
aws events put-targets --rule my-rule --targets
 Id=1,Arn=arn:aws:lambda:region:123456789012:function:my-function
```

En el comando anterior, *my-rule* aparece el nombre que especificó para la regla en el paso 2 y el valor del `Arn` parámetro es el ARN de la función que creó en el paso 1.

4. Para agregar permisos que permitan a la regla invocar en Lambda de destino, utilice el siguiente comando de Lambda add-permission. Este comando confía en el principal del EventBridge servicio (`events.amazonaws.com`) y asigna los permisos a la regla especificada.

```
aws lambda add-permission --function-name my-function --statement-id my-unique-id \
  --action 'lambda:InvokeFunction' --principal events.amazonaws.com --source-arn
 arn:aws:events:region:123456789012:rule/my-rule
```

En el comando anterior:

- *my-function*es el nombre de la función de Lambda que desea que la regla utilice como objetivo.
- *my-unique-id*es un identificador único que se define para describir la sentencia en la política de funciones de Lambda.
- `source-arn`es el ARN de la EventBridge regla.

Si el comando se ejecuta correctamente, verá un resultado similar al siguiente.

```
{
  "Statement": "{\"Sid\":\"my-unique-id\",
    \"Effect\":\"Allow\",
    \"Principal\":{\"Service\":\"events.amazonaws.com\"},
    \"Action\":\"lambda:InvokeFunction\",
    \"Resource\":\"arn:aws:lambda:us-west-2:123456789012:function:my-function\",
```

```
    \"Condition\":
      {\"ArnLike\":
        {\"AWS:SourceArn\":
         \"arn:aws:events:us-west-2:123456789012:rule/my-rule\"}}}"
}
```

El valor de `Statement` es una versión de cadena JSON de la instrucción que se agregó a la política de la función Lambda.

5. Una vez que haya seguido estas instrucciones, vaya a Incorporación de los enlaces de ciclo de vida en su grupo de escalado automático como siguiente paso.

Recepción de notificaciones mediante Amazon SNS

Puede utilizar Amazon SNS para configurar un destino de notificación (un tema de SNS) para recibir notificaciones cuando se produzca una acción del ciclo de vida. Luego, Amazon SNS envía las notificaciones a los destinatarios suscritos. Hasta que no se confirme la suscripción, no se enviarán a los destinatarios las notificaciones publicadas en el tema.

Para configurar notificaciones mediante Amazon SNS

1. Cree un tema de Amazon SNS mediante la consola de Amazon SNS o el comando create-topic. Asegúrese de que el tema se encuentre en la misma región que el grupo de Auto Scaling que está utilizando. Para obtener más información, consulte Introducción a Amazon SNS en la Guía para desarrolladores de Amazon Simple Notification Service.

```
aws sns create-topic --name my-sns-topic
```

2. Anote el nombre de recurso de Amazon (ARN) del tema, por ejemplo, `arn:aws:sns:region:123456789012:my-sns-topic`. Lo necesita para crear el enlace de ciclo de vida.
3. Cree una función de servicio de IAM para dar a Amazon EC2 Auto Scaling acceso a su destino de notificación de Amazon SNS.

   Para dar a Amazon EC2 Auto Scaling acceso a su tema de SNS

   a. Abra la consola de IAM en https://console.aws.amazon.com/iam/.
   b. En el panel de navegación de la izquierda, seleccione Roles.
   c. Elija Crear rol.

d. En Select trusted entity (Seleccionar entidad de confianza), elija AWS service (Servicio de).

e. Para su caso de uso, en Use cases for other AWS services (Casos de uso para otros servicios de), elija EC2 Auto Scaling y, luego, EC2 Auto Scaling Notification Access (Acceso a notificaciones de EC2 Auto Scaling).

f. Elija Next (Siguiente) dos veces para ir a la página Name, review, and create (Asignar nombre, revisar y crear).

g. En Role name (Nombre de rol), ingrese un nombre para el rol, (por ejemplo, **`my-notification-role`**) y elija Create role (Crear rol).

h. En la página Roles, elija el rol que acaba de crear para abrir la página Summary (Resumen). Anote el ARN del rol. Por ejemplo, `arn:aws:iam::`*`123456789012`*`:role/`*`my-notification-role`*. Lo necesita para crear el enlace de ciclo de vida.

4. Una vez que haya seguido estas instrucciones, vaya a Añada ganchos de ciclo de vida (AWS CLI) como siguiente paso.

Recepción de notificaciones mediante Amazon SQS

Puede utilizar Amazon SQS para configurar un destino de notificación que reciba mensajes cuando se produzca una acción del ciclo de vida. A continuación, un consumidor de colas debe sondear una cola de SQS para actuar sobre estas notificaciones.

> **Important**
>
> Las colas FIFO no son compatibles con enlaces de ciclo de vida.

Para configurar notificaciones mediante Amazon SQS

1. Cree una cola de Amazon SQS mediante la consola de Amazon SQS. Asegúrese de que la cola se encuentre en la misma región que el grupo de Auto Scaling que está utilizando. Para obtener más información, consulte Introducción a Amazon SQS en la Guía para desarrolladores de Amazon Simple Queue Service.
2. Anote el ARN de la cola, por ejemplo, `arn:aws:sqs:`*`us-west-2`*`:`*`123456789012`*`:`*`my-sqs-queue`*. Lo necesita para crear el enlace de ciclo de vida.
3. Cree una función de servicio de IAM para dar a Amazon EC2 Auto Scaling acceso a su destino de notificación de Amazon SQS.

Para dar a Amazon EC2 Auto Scaling acceso a su cola de SQS

a. Abra la consola de IAM en https://console.aws.amazon.com/iam/.
b. En el panel de navegación de la izquierda, seleccione Roles.
c. Elija Crear rol.
d. En Select trusted entity (Seleccionar entidad de confianza), elija AWS service (Servicio de).
e. Para su caso de uso, en Use cases for other AWS services (Casos de uso para otros servicios de), elija EC2 Auto Scaling y, luego, EC2 Auto Scaling Notification Access (Acceso a notificaciones de EC2 Auto Scaling).
f. Elija Next (Siguiente) dos veces para ir a la página Name, review, and create (Asignar nombre, revisar y crear).
g. En Role name (Nombre de rol), ingrese un nombre para el rol, (por ejemplo, **`my-notification-role`**) y elija Create role (Crear rol).
h. En la página Roles, elija el rol que acaba de crear para abrir la página Summary (Resumen). Anote el ARN del rol. Por ejemplo, `arn:aws:iam::`*`123456789012`*`:role/`*`my-notification-role`*. Lo necesita para crear el enlace de ciclo de vida.

4. Una vez que haya seguido estas instrucciones, vaya a Añada ganchos de ciclo de vida (AWS CLI) como siguiente paso.

Dirige las notificaciones a AWS Lambda directamente

Puede utilizar una función de Lambda como destino de notificación cuando se produzca una acción del ciclo de vida.

Para enviar las notificaciones a AWS Lambda directamente

1. Abra la página de Funciones en la consola de Lambda.
2. Elija la función de Lambda deseada.

   Para crear una nueva función de Lambda, consulte Crear la función de Lambda
3. Seleccione la pestaña Configuration (Configuración) y luego Permissions (Permisos).
4. Desplázate hacia abajo hasta Resource-based la política y, a continuación, selecciona Agregar permisos. Una política basada en recursos se utiliza para conceder permisos para invocar la función a la entidad principal que se especifica en la política. En este caso, la entidad principal

será el rol vinculado al servicio de Amazon EC2 Auto Scaling asociado al grupo de escalado automático.

5. En la sección Policy statement (Instrucción de la política), configure sus permisos:

   a. Elija Cuenta de AWS.

   b. En Principal (Entidad principal), escriba el ARN del rol vinculado al servicio de llamada, por ejemplo, **`arn:aws:iam::<aws-account-id>:role/aws-service-role/autoscaling.amazonaws.com/AWSServiceRoleForAutoScaling`**.

   c. En Action, elija lambda: InvokeFunction.

   d. En Statement ID (ID de estado), escriba un ID de instrucción único, como **`AllowInvokeByAutoScaling`**.

   e. Seleccione Save.

6. Una vez que haya seguido estas instrucciones, vaya a Añada ganchos de ciclo de vida (AWS CLI) como siguiente paso.

Ejemplo de mensaje de notificación

En esta sección se proporciona un ejemplo de notificación para Amazon SNS, Amazon SQS y. AWS Lambda

Mientras la instancia está en estado de espera, se publica un mensaje en Amazon SNS, Amazon SQS y en el destino de notificaciones. AWS Lambda

El mensaje incluye la siguiente información:

- `Origin`: un lugar del que proviene una instancia de EC2.
- `Destination`: un lugar al que va a ir una instancia de EC2.
- `LifecycleActionToken`: el token de acción del ciclo de vida.
- `AccountId`— El Cuenta de AWS ID.
- `AutoScalingGroupName`: el nombre del grupo de Auto Scaling.
- `LifecycleHookName`: el nombre del enlace de ciclo de vida.
- `EC2InstanceId`: el ID de la instancia EC2.
- `LifecycleTransition`: el tipo de enlace de ciclo de vida.
- `NotificationMetadata`: los metadatos de la notificación.

A continuación, se muestra un ejemplo de mensaje de notificación.

```
Service: AWS Auto Scaling
Time: 2021-01-19T00:36:26.533Z
RequestId: 18b2ec17-3e9b-4c15-8024-ff2e8ce8786a
Origin: EC2
Destination: AutoScalingGroup
LifecycleActionToken: 71514b9d-6a40-4b26-8523-05e7ee35fa40
AccountId: 123456789012
AutoScalingGroupName: my-asg
LifecycleHookName: my-hook
EC2InstanceId: i-0598c7d356eba48d7
LifecycleTransition: autoscaling:EC2_INSTANCE_LAUNCHING
NotificationMetadata: hook message metadata
```

Ejemplo de mensaje de notificación de prueba

Al agregar por primera vez un enlace de ciclo de vida, un mensaje de notificación de prueba se publica en el destino de notificación. A continuación, se muestra un ejemplo de mensaje de notificación de prueba.

```
Service: AWS Auto Scaling
Time: 2021-01-19T00:35:52.359Z
RequestId: 18b2ec17-3e9b-4c15-8024-ff2e8ce8786a
Event: autoscaling:TEST_NOTIFICATION
AccountId: 123456789012
AutoScalingGroupName: my-asg
AutoScalingGroupARN: arn:aws:autoscaling:us-
west-2:123456789012:autoScalingGroup:042cba90-
ad2f-431c-9b4d-6d9055bcc9fb:autoScalingGroupName/my-asg
```

> **Note**
>
> Para ver ejemplos de los eventos enviados por Amazon EC2 Auto Scaling a EventBridge, consulteReferencia de evento de Amazon EC2 Auto Scaling.

Controle la retención de instancias con políticas de ciclo de vida de las instancias

Las políticas del ciclo de vida de las instancias brindan protección contra las terminaciones de Amazon EC2 Auto Scaling cuando se abandona una acción del ciclo de vida de terminación. A diferencia de los ganchos del ciclo de vida, las políticas del ciclo de vida de las instancias están diseñadas para garantizar que las instancias pasen a un estado de retención cuando los procedimientos de cierre correctos no se completen correctamente.

¿Cuándo usar las políticas del ciclo de vida de las instancias

Usa las políticas del ciclo de vida de las instancias cuando el cierre correcto de la aplicación no sea opcional, sino obligatorio, y los cierres fallidos requieran una intervención manual. Los casos de uso comunes incluyen:

- Aplicaciones con estado que deben completar la persistencia de los datos antes de su finalización.
- Aplicaciones que requieren períodos de agotamiento prolongados que pueden superar el tiempo máximo de espera de 48 horas del ciclo de vida de los enganches.
- Cargas de trabajo que gestionan datos confidenciales en las que una limpieza fallida o incompleta podría provocar la pérdida o corrupción de los datos.
- Mission-critical servicios en los que un cierre brusco repercute en la disponibilidad.

Para obtener más información sobre cómo gestionar correctamente la terminación de instancias, consulte. Diseño de aplicaciones para gestionar sin problemas la terminación de instancias

Cómo funcionan las políticas del ciclo de vida de las instancias con los ganchos del ciclo de vida

Las políticas del ciclo de vida de las instancias funcionan en combinación con los ganchos del ciclo de vida de terminación, no como sustitutos. El proceso consta de varias etapas:

1. Se ejecutan las acciones del ciclo de vida de terminación. Cuando Amazon EC2 Auto Scaling selecciona una instancia para su terminación, se invocan los enlaces del ciclo de vida de terminación y la instancia entra en el `Terminating:Wait` estado para comenzar a ejecutar las acciones del ciclo de vida de terminación.
2. Comienza un intento de apagado correcto. La aplicación, que se ejecuta en la instancia o mediante un plano de control, recibe la notificación de la acción de finalización del ciclo de vida e

inicia los procedimientos de cierre correctos, como agotar las conexiones, completar el trabajo en curso o transferir datos.

3. Se han completado las acciones del ciclo de vida de terminación Una acción del ciclo de vida de terminación puede completarse `CONTINUE` o `ABANDON` resultar en ella.
4. La política del ciclo de vida de la instancia evalúa la situación. Sin configurar una política de ciclo de vida de la instancia, la instancia finaliza de inmediato, incluso si la acción del ciclo de vida de terminación se completó con un `ABANDON` resultado. Con una política del ciclo de vida de la instancia configurada para retener las instancias activadas`TerminateHookAbandon`, la instancia pasa a un estado de retención si la acción de terminación del ciclo de vida se completó con `ABANDON` resultados.
5. Las instancias retenidas están a la espera de una acción manual. Las instancias en estados retenidos siguen incurriendo en cargos estándar de Amazon EC2. Estas instancias no se tienen en cuenta para la capacidad deseada de su grupo de Auto Scaling, por lo que Auto Scaling lanza instancias de reemplazo para mantener el tamaño deseado. Las funciones de Auto Scaling, como la actualización de instancias y la duración máxima de las instancias, también ignorarán las instancias retenidas. Esto te permite completar los procedimientos de limpieza de forma manual, recuperar datos o investigar por qué falló el apagado automático antes de terminar la instancia de forma manual.
6. Se produce la terminación manual. Después de completar las acciones necesarias en la instancia retenida, debes llamar a la `TerminateInstanceInAutoScalingGroup` API para terminar la instancia.

Configura la retención de instancias

Configure su grupo de Amazon EC2 Auto Scaling para retener las instancias cuando no se cumplan las acciones de finalización del ciclo de vida.

Para usar las políticas del ciclo de vida de las instancias en su grupo de Auto Scaling, también debe configurar un enlace al ciclo de vida de terminación. Si configuras una política del ciclo de vida de una instancia pero no tienes ningún enlace al ciclo de vida de terminación, la política no tiene ningún efecto. Las políticas del ciclo de vida de las instancias solo se aplicarán cuando se abandonen las acciones del ciclo de vida de terminación, no cuando se completen correctamente con el `CONTINUE` resultado.

Las políticas del ciclo de vida de las instancias utilizan activadores de retención para determinar cuándo retener una instancia. El `TerminateHookAbandon` desencadenante provoca la retención en varios escenarios:

- Cuando llamas explícitamente a la CompleteLifecycleAction API con el ABANDON resultado.
- Cuando se agota el ABANDON tiempo de espera de una acción del ciclo de vida de terminación con un resultado predeterminado porque se ha agotado el tiempo de espera sin recibir ningún latido.
- Cuando se alcanza el tiempo de espera global de una acción del ciclo de vida de terminación con el resultado predeterminadoABANDON, que es de 48 horas o 100 veces el tiempo de espera del latido, lo que sea menor

Console

Para configurar la retención de instancias

1. Abra la consola Amazon EC2 Auto Scaling
2. Cree su grupo de Auto Scaling (la política del ciclo de vida de las instancias pasa por defecto a Terminar)
3. Ve a la página de detalles del grupo de Auto Scaling y elige la pestaña Administración de instancias
4. En la política del ciclo de vida de las instancias para ver los ganchos del ciclo de vida, elige
5. Crea tus ganchos del ciclo de vida de terminación con:
   - La transición del ciclo de vida está configurada para terminar la instancia
   - El resultado predeterminado está establecido en Abandon

AWS CLI

Para configurar la retención de instancias

Usa el comando create-auto-scaling-group con una política de ciclo de vida de la instancia:

```
aws autoscaling create-auto-scaling-group \
--auto-scaling-group-name my-asg \
--launch-template LaunchTemplateName=my-template,Version='$Latest' \
--min-size 1 \
--max-size 3 \
--desired-capacity 2 \
--vpc-zone-identifier subnet-12345678 \
--instance-lifecycle-policy file://lifecycle-policy.json
```

Contenido de lifecycle-policy.json:

```
{
    "RetentionTriggers": {
        "TerminateHookAbandon": "retain"
    }
}
```

Para agregar un enlace de ciclo de vida de terminación

Utilice el comando put-lifecycle-hook:

```
aws autoscaling put-lifecycle-hook \
--lifecycle-hook-name my-termination-hook \
--auto-scaling-group-name my-asg \
--lifecycle-transition autoscaling:EC2_INSTANCE_TERMINATING \
--default-result ABANDON \
--heartbeat-timeout 300
```

Gestione las instancias retenidas

Supervise y controle las instancias de Amazon EC2 que se han trasladado a un estado retenido. Utilice CloudWatch las métricas para realizar un seguimiento de las instancias retenidas y, a continuación, finalice manualmente las instancias retenidas tras completar las acciones personalizadas.

Las instancias retenidas no se tienen en cuenta para la capacidad deseada de su grupo Amazon EC2 Auto Scaling. Cuando una instancia entra en un estado de retención, Auto Scaling lanza una instancia de reemplazo para mantener la capacidad deseada. Por ejemplo, supongamos que su grupo de Auto Scaling tiene una capacidad deseada de 10. Cuando una instancia entra en `Terminating:Retained` estado, Auto Scaling lanza una instancia de reemplazo para mantener la capacidad deseada de 10. Ahora tiene 11 instancias en ejecución en total: 10 en su grupo activo y 1 instancia retenida. Se aplicarán los cargos estándar de Amazon EC2 a las 11 instancias hasta que cancele manualmente la instancia retenida.

Estados del ciclo de vida de las instancias retenidas

Comprenda cómo las instancias pasan por los estados del ciclo de vida cuando se utilizan las políticas del ciclo de vida de las instancias. Las instancias siguen una ruta específica desde la terminación normal, pasando por la retención, hasta la terminación final.

Cuando se activa la retención, las instancias pasan por estos estados:

1. `Terminating`- Comienza la terminación normal
2. `Terminating:Wait`- Se ejecuta el enlace Lifecycle
3. `Terminating:Proceed`- Concluyen las acciones del ciclo de vida (independientemente de si tuvieron éxito o no)
4. `Terminating:Retained`- El enlace falla, la instancia se conserva para una intervención manual

Las instancias de Warm Pool adoptan diferentes rutas de estado de ciclo de vida según el escenario:

Instancias que vuelven a convertirse en un grupo activo:

1. `Warmed:Pending`- Comienza la transición normal a una piscina caliente
2. `Warmed:Pending:Wait`- Se ejecuta el enlace Lifecycle
3. `Warmed:Pending:Proceed`- Concluyen las acciones del ciclo de vida (independientemente de si tuvieron éxito o no)
4. `Warmed:Pending:Retained`- El enlace falla, la instancia se conserva para una intervención manual

Instancias que se están cancelando desde la piscina caliente:

1. `Warmed:Terminating`- Comienza la terminación normal
2. `Warmed:Terminating:Wait`- Se ejecuta el enlace Lifecycle
3. `Warmed:Terminating:Proceed`- Concluyen las acciones del ciclo de vida (independientemente de si tuvieron éxito o no)
4. `Warmed:Terminating:Retained`- El enlace falla, la instancia se conserva para una intervención manual

Supervise las instancias retenidas

Dado que las instancias de Amazon EC2 retenidas conllevan costes y requieren una intervención manual, es fundamental supervisarlas. Amazon EC2 Auto Scaling proporciona varias CloudWatch métricas para realizar un seguimiento de las instancias retenidas.

Habilite las métricas de grupo para realizar un seguimiento de las instancias retenidas:

```
aws autoscaling enable-metrics-collection \
--auto-scaling-group-name my-asg \
--metrics GroupTerminatingRetainedInstances
```

Las métricas disponibles son:

- `GroupTerminatingRetainedInstances`muestra el número de instancias del `Terminating:Retained` estado.
- `GroupTerminatingRetainedCapacity`muestra las unidades de capacidad representadas por las instancias del `Terminating:Retained` estado.
- `WarmPoolTerminatingRetainedCapacity`rastrea las instancias retenidas que terminan en la piscina caliente.
- `WarmPoolPendingRetainedCapacity`rastrea las instancias retenidas que regresan a la piscina caliente.

También puede consultar las actividades de escalado de su grupo de Auto Scaling de Amazon EC2 para saber por qué se retuvieron las instancias. Busque las actividades de finalización con mensajes de motivos de estado que indiquen errores en los enlaces del ciclo de vida `StatusCode: Cancelled` y los motivos de su estado:

```
aws autoscaling describe-scaling-activities \
--auto-scaling-group-name my-asg
```

Te recomendamos crear CloudWatch alarmas basadas en estas métricas para avisarte cuando las instancias pasen a un estado de retención. Esto le ayuda a hacer un seguimiento de las implicaciones financieras y garantiza que no se olvide de limpiar las instancias que requieren una intervención manual.

Finalice las instancias retenidas

Tras completar tus acciones personalizadas, termina las instancias retenidas llamando a la TerminateInstanceInAutoScalingGroup API:

```
aws autoscaling terminate-instance-in-auto-scaling-group \
--instance-id i-1234567890abcdef0 \
--no-should-decrement-desired-capacity
```

Recuperar el estado de ciclo de vida de destino a través de los metadatos de instancia

Cada instancia de Auto Scaling que se lanza pasa por varios estados de ciclo de vida. Para invocar acciones personalizadas desde el interior de una instancia para que actúen en determinadas transiciones de estado de ciclo de vida, puede hacerlo recuperando el estado de ciclo de vida de destino a través de los metadatos de instancia.

Por ejemplo, es posible que necesite un mecanismo que detecte la terminación de la instancia desde dentro de la instancia para ejecutar parte del código en la instancia antes de que finalice. Para ello, escriba un código que sondee el estado del ciclo de vida de una instancia directamente desde la instancia. A continuación, puede agregar un enlace de ciclo de vida al grupo de escalado automático para mantener la instancia en ejecución hasta que el código envíe el comando complete-lifecycle-action para continuar.

El ciclo de vida de las instancias de Auto Scaling tiene dos estados estables principales (`InService` y `Terminated`) y dos estados estables secundarios (`Detached` y `Standby`). Si utiliza un grupo de calentamiento, el ciclo de vida tiene cuatro estados estables adicionales: `Warmed:Hibernated`, `Warmed:Running`, `Warmed:Stopped` y `Warmed:Terminated`.

Cuando una instancia se prepara para pasar a uno de los estados estables anteriores, Amazon EC2 Auto Scaling actualiza el valor del elemento `autoscaling/target-lifecycle-state` de los metadatos de instancia. Para obtener el estado de ciclo de vida de destino desde la propia instancia, debe utilizar el Servicio de metadatos de instancia para recuperarlo desde los metadatos de instancia.

> **Note**
>
> Los metadatos de instancia son datos de una instancia de Amazon EC2 que las aplicaciones pueden utilizar para consultar información sobre esa instancia. El Servicio de metadatos de instancia es un componente de la instancia que el código local utiliza para acceder a los metadatos de instancia. El código local puede incluir scripts de datos de usuario o aplicaciones que se ejecuten en la instancia.

El código local puede acceder a los metadatos de instancia de una instancia en ejecución mediante uno de estos dos métodos: Instance Metadata Service versión 1 (IMDSv1) o Instance Metadata Service versión 2 (IMDSv2). IMDSv2 utiliza solicitudes orientadas a la sesión y mitiga varios tipos

de vulnerabilidades que se podrían utilizar para intentar acceder a los metadatos de instancia. Para obtener más detalles sobre estos dos métodos, consulte Utilizar IMDSv2 en la Guía del usuario de Amazon EC2.

IMDSv2

```
[ec2-user ~]$ TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H "X-aws-ec2-metadata-token-ttl-seconds: 21600"` \
&& curl -H "X-aws-ec2-metadata-token: $TOKEN" -v http://169.254.169.254/latest/meta-data/autoscaling/target-lifecycle-state
```

IMDSv1

```
[ec2-user ~]$ curl http://169.254.169.254/latest/meta-data/autoscaling/target-lifecycle-state
```

A continuación, se muestra un ejemplo del resultado.

```
InService
```

El estado de ciclo de vida de destino es el estado al que va a pasar la instancia. El estado de ciclo de vida actual es el estado en el que se encuentra la instancia. Ambos pueden ser el mismo una vez que se complete la acción de ciclo de vida y la instancia finalice su paso al estado de ciclo de vida de destino. No se puede recuperar el estado de ciclo de vida actual de la instancia a partir de los metadatos de instancia.

Amazon EC2 Auto Scaling comenzó a generar el estado de ciclo de vida de destino el 10 de marzo de 2022. Si la instancia pasa a alguno de los estados de ciclo de vida de destino después de esa fecha, el elemento del estado de ciclo de vida de destino aparece en los metadatos de instancia. En caso contrario, no aparece, y verá un error HTTP 404.

Para obtener instrucciones acerca de la recuperación de metadatos de instancia, consulte Recuperar metadatos de instancia en la Guía del usuario de Amazon EC2.

Para ver un tutorial que muestra cómo crear un enlace de ciclo de vida con una acción personalizada en un script de datos de usuario que utiliza el estado de ciclo de vida de destino, consulte Tutorial: Uso del script de datos y los metadatos de la instancia para recuperar el estado del ciclo de vida.

> **Important**
>
> Para asegurarse de que puede invocar una acción personalizada lo antes posible, su código local debería sondear el IMDS con frecuencia y volver a intentarlo en caso de errores.

Incorporación de los enlaces de ciclo de vida en su grupo de escalado automático

Puede agregar enlaces del ciclo de vida al grupo de Auto Scaling para poner las instancias de Auto Scaling en estado de espera y llevar a cabo acciones personalizadas en ellas. Las acciones personalizadas se realizan a medida que se lanzan las instancias o antes de que finalicen. Las instancias permanecen en estado de espera hasta que se completa la acción del ciclo de vida o finaliza el periodo de espera.

Después de crear un grupo de Auto Scaling a partir de Consola de administración de AWS, puedes añadirle uno o más ganchos de ciclo de vida, hasta un total de 50 ganchos de ciclo de vida. También puedes usar el AWS CLI CloudFormation, o un SDK para añadir ganchos de ciclo de vida a un grupo de Auto Scaling mientras lo creas.

De forma predeterminada, al añadir un enlace de ciclo de vida a la consola, Amazon EC2 Auto Scaling envía notificaciones de eventos del ciclo de vida a Amazon. EventBridge Se recomienda utilizar un script de datos de usuario EventBridge o utilizar un script de datos de usuario. Para crear un enlace de ciclo de vida que envíe notificaciones directamente a Amazon SNS o Amazon SQS, también AWS Lambda puede utilizar el comando put-lifecycle-hook, como se muestra en los ejemplos de este tema.

Contenido

- Adición de enlaces de ciclo de vida (consola)
- Añada ganchos de ciclo de vida (AWS CLI)

Adición de enlaces de ciclo de vida (consola)

Siga estos pasos para agregar enlaces de ciclo de vida a un grupo de escalado automático. Para agregar enlaces de ciclo de vida a fin de escalar horizontalmente (lanzar instancias) y reducir horizontalmente (terminar instancias o regresarlas a un grupo en caliente), debe crear dos enlaces independientes.

Antes de comenzar, confirme que ha configurado una acción personalizada, según sea necesario, como se detalla en Preparación para agregar un enlace de ciclo de vida a un grupo de Auto Scaling.

Agregación de un enlace de ciclo de vida para escalar horizontalmente

1. Abra la consola de Amazon EC2 en y seleccione Auto Scaling Groups en el https://console.aws.amazon.com/ec2/ panel de navegación.
2. Seleccione la casilla situada junto al grupo de escalado automático. Se abre un panel dividido en la parte inferior de la página.
3. En la pestaña Instance management (Administración de instancias), en Lifecycle hooks (Enlaces de ciclo de vida), elija Create lifecycle hook (Crear enlace de ciclo de vida).
4. Para definir un enlace de ciclo de vida para escalar horizontalmente (lanzamiento de instancias), haga lo siguiente:

   a. En Lifecycle hook name (Nombre del enlace de ciclo de vida), especifique un nombre para el enlace de ciclo de vida.

   b. En Lifecycle transition (Transición del ciclo de vida), elija Instance launch (Lanzamiento de instancia).

   c. En Tiempo de espera del latido, especifique la cantidad de tiempo en segundos que las instancias pueden permanecer en estado de espera al escalar horizontalmente antes de que se agote el tiempo de espera del enlace. El rango va de `30` a `7200` segundos. Establecer un periodo de tiempo de espera prolongado proporciona más tiempo para que se complete la acción personalizada. Entonces, si termina antes de que finalice el período de espera, envíe el comando complete-lifecycle-action para permitir que la instancia pase al siguiente estado.

   d. En Default result (Resultado predeterminado), especifique la acción que se debe realizar cuando termine el tiempo de espera del enlace de ciclo de vida o cuando se produzca un error inesperado. Puede seleccionar CONTINUAR o ABANDONAR.

      - Si elige CONTINUAR, el grupo de escalado automático puede continuar con cualquier otro enlace de ciclo de vida y luego poner la instancia en servicio.
      - Si elige ABANDONAR, el grupo de escalado automático detiene las acciones restantes y termina las instancias de inmediato.

   e. (Opcional) En Metadatos de notificación, especifique cualquier otra información que desee incluir cuando Amazon EC2 Auto Scaling envíe un mensaje al destino de notificación.

5. Seleccione Crear.

Agregación de un enlace de ciclo de vida para reducir horizontalmente

1. Elija Crear enlace de ciclo de vida para continuar donde lo dejó después de crear un enlace de ciclo de vida para escalar horizontalmente.
2. Para definir un enlace de ciclo de vida para reducir horizontalmente (instancias que finalizan o regresan a un grupo de calentamiento), haga lo siguiente:

   a. En Lifecycle hook name (Nombre del enlace de ciclo de vida), especifique un nombre para el enlace de ciclo de vida.
   b. En Lifecycle transition (Transición del ciclo de vida), elija Instance terminate (Terminación de instancia).
   c. En Tiempo de espera del latido, especifique la cantidad de tiempo en segundos que las instancias pueden permanecer en estado de espera al escalar horizontalmente antes de que se agote el tiempo de espera del enlace. Se recomienda un breve período de tiempo de espera de `30` `120` unos segundos, en función del tiempo que necesite para realizar cualquier tarea final, como extraer los registros de EC2. CloudWatch
   d. En Default result (Resultado predeterminado), especifique la acción que el grupo de escalado automático va a realizar cuando transcurra el tiempo de espera o si se produce un error inesperado. Tanto ABANDON (Abandonar) como CONTINUE (Continuar) permiten que la instancia se termine.

      - Si elige CONTINUE (Continuar), el grupo de escalado automático puede continuar con todas las acciones restantes, como otros enlaces de ciclo de vida, antes de la terminación.
      - Si elige ABANDONAR, el grupo de escalado automático termina la instancia de inmediato.

   e. (Opcional) En Metadatos de notificación, especifique cualquier otra información que desee incluir cuando Amazon EC2 Auto Scaling envíe un mensaje al destino de notificación.
3. Seleccione Crear.

Añada ganchos de ciclo de vida (AWS CLI)

Cree y actualice enlaces de ciclo de vida mediante el comando put-lifecycle-hook.

Para realizar una acción de escalado ascendente, utilice el siguiente comando.

```
aws autoscaling put-lifecycle-hook --lifecycle-hook-name my-launch-hook  \
```

```
  --auto-scaling-group-name my-asg \
  --lifecycle-transition autoscaling:EC2_INSTANCE_LAUNCHING
```

Para realizar una acción de reducción horizontal, utilice el siguiente comando.

```
aws autoscaling put-lifecycle-hook --lifecycle-hook-name my-termination-hook  \
  --auto-scaling-group-name my-asg \
  --lifecycle-transition autoscaling:EC2_INSTANCE_TERMINATING
```

Para recibir notificaciones mediante Amazon SNS o Amazon SQS, agregue las opciones `--notification-target-arn` y `--role-arn`. Para recibir notificaciones mediante AWS Lambda, añada el`--notification-target-arn`.

En el siguiente ejemplo, se crea un enlace de ciclo de vida que especifica un tema de SNS denominado *my-sns-topic* como destino de notificación.

```
aws autoscaling put-lifecycle-hook --lifecycle-hook-name my-termination-hook  \
  --auto-scaling-group-name my-asg \
  --lifecycle-transition autoscaling:EC2_INSTANCE_TERMINATING \
  --notification-target-arn arn:aws:sns:region:123456789012:my-sns-topic \
  --role-arn arn:aws:iam::123456789012:role/my-notification-role
```

El tema recibe una notificación de prueba con el siguiente par de clave-valor.

```
"Event": "autoscaling:TEST_NOTIFICATION"
```

De manera predeterminada, el comando put-lifecycle-hook crea un enlace de ciclo de vida con un tiempo de espera de latido de `3600` segundos (una hora).

Para cambiar el tiempo de espera de latido de un enlace de ciclo de vida existente, agregue la opción `--heartbeat-timeout`, como se muestra en el siguiente ejemplo.

```
aws autoscaling put-lifecycle-hook --lifecycle-hook-name my-termination-hook \
  --auto-scaling-group-name my-asg --heartbeat-timeout 120
```

Si una instancia ya se encuentra en estado de espera, puede evitar que el enlace de ciclo de vida agote el tiempo de espera registrando un latido con el comando record-lifecycle-action-heartbeat de la CLI. De esta forma, se incrementa el tiempo de espera en el valor especificado cuando creó el enlace de ciclo de vida. Si termina antes de que finalice el periodo de tiempo de espera, puede enviar el comando complete-lifecycle-action de la CLI para permitir que la instancia pase al siguiente

estado. Para obtener más información y ejemplos, consulta Cómo completar una acción del ciclo de vida en un grupo de escalado automático.

Cómo completar una acción del ciclo de vida en un grupo de escalado automático

Cuando un grupo de Auto Scaling responde a un evento del ciclo de vida, pone a la instancia en estado de espera y envía una notificación del evento. Mientras la instancia se encuentra en estado de espera, puede realizar una acción personalizada.

Resulta útil completar la acción del ciclo de vida con un resultado de `CONTINUE` si se termina antes de que venza el tiempo de espera. Si no completa la acción del ciclo de vida, el enlace de ciclo de vida pasa al estado que especificó para Resultado predeterminado una vez finalizado el período de tiempo de espera.

Contenido

- Completar una acción del ciclo de vida (manual)
- Completar una acción del ciclo de vida (automático)

Completar una acción del ciclo de vida (manual)

El siguiente procedimiento corresponde a la interfaz de línea de comandos y no se admite en la consola. La información que debe reemplazarse, como el ID de la instancia o el nombre de un grupo de Auto Scaling, aparece en cursiva.

Para completar una acción del ciclo de vida (AWS CLI)

1. Si necesita más tiempo para completar la acción personalizada, utilice el comando record-lifecycle-action-heartbeat para reiniciar el periodo de tiempo de espera y mantener la instancia en estado de espera. Por ejemplo, si el periodo de tiempo de espera es una hora y llama a este comando después de 30 minutos, la instancia permanece en estado de espera durante una hora más, es decir, 90 minutos en total.

   Puede especificar el token de acción del ciclo de vida que recibió con la notificación, como se muestra en el siguiente comando.

   ```
   aws autoscaling record-lifecycle-action-heartbeat --lifecycle-hook-name my-launch-hook \
   ```

```
  --auto-scaling-group-name my-asg --lifecycle-action-
token bcd2f1b8-9a78-44d3-8a7a-4dd07d7cf635
```

También puede especificar el ID de la instancia que recibió con la notificación, como se muestra en el siguiente comando.

```
aws autoscaling record-lifecycle-action-heartbeat --lifecycle-hook-name my-launch-
hook \
  --auto-scaling-group-name my-asg --instance-id i-1a2b3c4d
```

2. Si termina la acción personalizada antes de que finalice el tiempo de espera, utilice el comando complete-lifecycle-action para que el grupo de Auto Scaling pueda continuar iniciando o terminando la instancia. Puede especificar el token de acción del ciclo de vida, tal y como se muestra en el siguiente comando.

```
aws autoscaling complete-lifecycle-action --lifecycle-action-result CONTINUE \
  --lifecycle-hook-name my-launch-hook --auto-scaling-group-name my-asg \
  --lifecycle-action-token bcd2f1b8-9a78-44d3-8a7a-4dd07d7cf635
```

También puede especificar el ID de la instancia, tal y como se muestra en el siguiente comando.

```
aws autoscaling complete-lifecycle-action --lifecycle-action-result CONTINUE \
  --instance-id i-1a2b3c4d --lifecycle-hook-name my-launch-hook \
  --auto-scaling-group-name my-asg
```

Completar una acción del ciclo de vida (automático)

Si tiene un script de datos de usuario que configura las instancias después del lanzamiento, no es necesario que complete las acciones del ciclo de vida de forma manual. Puede agregar el comando complete-lifecycle-action al script. El script puede recuperar el ID de instancia de los metadatos de instancia e indicar a Amazon EC2 Auto Scaling cuando los scripts de arranque se hayan completado con éxito.

Si aún no lo ha hecho, actualice el script para recuperar el ID de instancia de la instancia de los metadatos de instancia. Para obtener más información, consulte Recuperar metadatos de instancia en la Guía del usuario de Amazon EC2.

Si utiliza Lambda, también puede configurar una devolución de llamada en el código de su función para permitir que el ciclo de vida de la instancia continúe si la acción personalizada se realiza

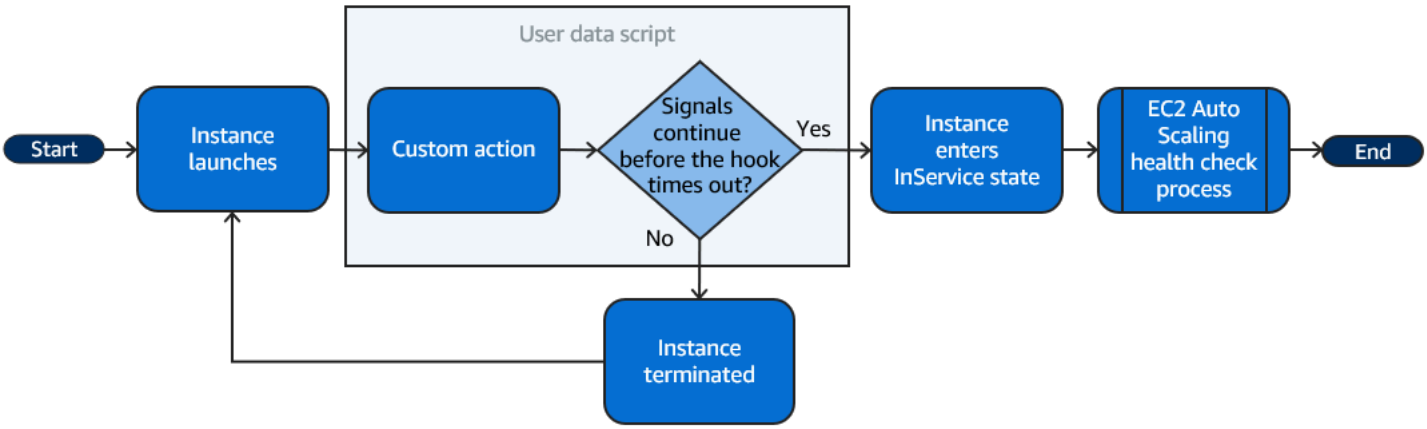
correctamente. Para obtener más información, consulte Tutorial: Configuración de un enlace de ciclo de vida que invoca una función Lambda.

Tutorial: Uso del script de datos y los metadatos de la instancia para recuperar el estado del ciclo de vida

Una forma habitual de crear acciones personalizadas para los enlaces del ciclo de vida es utilizar las notificaciones que Amazon EC2 Auto Scaling envía a otros servicios, como Amazon EventBridge. No obstante, puede evitar tener que crear infraestructura adicional si, en lugar de eso, utiliza un script de datos de usuario para trasladar el código que configura las instancias y completa la acción de ciclo de vida a las propias instancias.

En el siguiente tutorial, se muestran los primeros pasos para utilizar un script de datos de usuario y los metadatos de instancia. Puede crear una configuración de grupo de Auto Scaling básica con un script de datos de usuario que lea el estado de ciclo de vida de destino de las instancias de un grupo y realice una acción de devolución de llamada en una fase específica del ciclo de vida de una instancia para continuar con el proceso de lanzamiento.

En la siguiente ilustración se resume el flujo de un evento de escalado horizontal cuando utilice un script de datos de usuario para realizar una acción personalizada. Después de la inicialización de una instancia, el ciclo de vida de la instancia se detiene hasta que se complete el enlace de ciclo de vida, ya sea porque se agota el tiempo de espera o porque Amazon EC2 Auto Scaling recibe una señal para continuar.



Contenido

- Paso 1: Crear un rol de IAM con permisos para completar acciones de ciclo de vida
- Paso 2: Crear una plantilla de lanzamiento e incluir el rol de IAM y un script de datos de usuario
- Paso 3: Crear un grupo de Auto Scaling

- Paso 4: agregar un enlace de ciclo de vida
- Paso 5: Probar y verificar la funcionalidad
- Paso 6: limpiar
- Recursos relacionados

Paso 1: Crear un rol de IAM con permisos para completar acciones de ciclo de vida

Cuando utilices el SDK AWS CLI o un AWS SDK para enviar una llamada con el fin de completar las acciones del ciclo de vida, debes usar un rol de IAM con permisos para completar las acciones del ciclo de vida.

Para crear la política de

1. En la consola de IAM, abra la página Policies (Políticas), y, a continuación, elija Create policy (Crear política).
2. Seleccione la pestaña JSON.
3. En el cuadro Policy Document (Documento de política), copie y pegue el siguiente documento de política. Sustituye el *sample text* por tu número de cuenta y el nombre del grupo de Auto Scaling que quieres crear (). **TestAutoScalingEvent-group**

   JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "autoscaling:CompleteLifecycleAction"
      ],
      "Resource":
 "arn:aws:autoscaling:*:123456789012:autoScalingGroup:*:autoScalingGroupName/
TestAutoScalingEvent-group"
    }
  ]
}
```

4. Elija Siguiente.

5. Para Nombre de la política, introduzca **`TestAutoScalingEvent-policy`**. Elija Crear política.

Cuando termine de crear la política, podrá crear un rol que la utilice.

Para crear el rol.

1. En el panel de navegación de la izquierda, seleccione Roles.
2. Elija Crear rol.
3. En Select trusted entity (Seleccionar entidad de confianza), elija AWS service (Servicio de).
4. Para el caso de uso, elija EC2 y, luego, Next (Siguiente).
5. En Agregar permisos, elige la política que creaste (TestAutoScalingEvent-policy). A continuación, elija Siguiente.
6. En la página Name, review and create (Asignar nombre, revisar y crear), para Role name (Nombre del rol), ingrese **`TestAutoScalingEvent-role`** y seleccione Create role (Crear rol).

Paso 2: Crear una plantilla de lanzamiento e incluir el rol de IAM y un script de datos de usuario

Cree una plantilla de lanzamiento para usarla con un grupo de Auto Scaling. Incluya el rol de IAM que ha creado y el script de datos de usuario de ejemplo proporcionado.

Para crear una plantilla de lanzamiento

1. Abra la página Launch templates (Plantillas de lanzamiento) de la consola de Amazon EC2.
2. Elija Crear plantilla de inicialización.
3. Para Launch template name (Nombre de plantilla de lanzamiento), ingrese **`TestAutoScalingEvent-template`**.
4. En Auto Scaling guidance (Guía de Auto Scaling), seleccione la casilla de verificación.
5. En Application and OS Images (Amazon Machine Image) (Imágenes de aplicaciones y SO [imagen de máquina de Amazon]), elija Amazon Linux 2 (HVM), SSD Volume Type, 64-bit (x86) (Amazon Linux 2 [HVM], tipo de volumen SSD, 64 bits [x86]) en la lista de Quick Start (Inicio rápido).
6. En Instance type (Tipo de instancia), elija un tipo de instancia de Amazon EC2 (por ejemplo, “t2.micro”).
7. Para Advanced details (Detalles avanzados), expanda la sección para ver los campos.

8. Para el perfil de instancia de IAM, elige el nombre del perfil de instancia de IAM de tu rol de IAM (). TestAutoScalingEvent-role Un perfil de instancias es un contenedor de un rol de IAM que permite que Amazon EC2 transfiera el rol de IAM a una instancia cuando esta se lanza.

   Cuando se utilizaba la consola de IAM para crear un rol de IAM, esta creaba automáticamente un perfil de instancia con el mismo nombre que el rol correspondiente.

9. Copie y pegue el siguiente script de datos de usuario de ejemplo en el campo User data (Datos de usuario). Sustituya el texto de ejemplo de por `group_name` el nombre del grupo de Auto Scaling que quiere crear y `region` por el que quiere que Región de AWS utilice su grupo de Auto Scaling.

```
#!/bin/bash

function token {
    echo "X-aws-ec2-metadata-token: $(curl -X PUT 'http://169.254.169.254/latest/
api/token' -H 'X-aws-ec2-metadata-token-ttl-seconds: 21600')"
}

function get_target_state {
    echo $(curl -H "$(token)" -s http://169.254.169.254/latest/meta-data/
autoscaling/target-lifecycle-state)
}

function get_instance_id {
    echo $(curl -H "$(token)" -s http://169.254.169.254/latest/meta-data/instance-
id)
}

function complete_lifecycle_action {
    instance_id=$(get_instance_id)
    group_name='TestAutoScalingEvent-group'
    region='us-west-2'

    echo $instance_id
    echo $region
    echo $(aws autoscaling complete-lifecycle-action \
      --lifecycle-hook-name TestAutoScalingEvent-hook \
      --auto-scaling-group-name $group_name \
      --lifecycle-action-result CONTINUE \
      --instance-id $instance_id \
      --region $region)
}
```

```
function main {
    while true
    do
        target_state=$(get_target_state)
        if [ \"$target_state\" = \"InService\" ]; then
            # Change hostname
            export new_hostname="${group_name}-$instance_id"
            hostname $new_hostname
            # Send callback
            complete_lifecycle_action
            break
        fi
        echo $target_state
        sleep 5
    done
}

main
```

Este sencillo script de datos de usuario hace lo siguiente:

- Llama a los metadatos de instancia para recuperar el estado de ciclo de vida de destino y el ID de instancia de los metadatos de instancia.
- Recupera el estado de ciclo de vida de destino reiteradamente hasta que cambia a `InService`.
- Cambia el nombre de host de la instancia por el ID de instancia precedido del nombre del grupo de Auto Scaling, si el estado de ciclo de vida de destino es `InService`.
- Envía una devolución de llamada llamando al comando complete-lifecycle-action de la CLI para indicar a Amazon EC2 Auto Scaling que debe `CONTINUE` (continuar) con el proceso de lanzamiento de EC2.

10. Elija Crear plantilla de inicialización.
11. En la página de confirmación, seleccione Create Auto Scaling group (Crear grupo de Auto Scaling.

> **Note**
>
> Para ver otros ejemplos que pueda usar como referencia para desarrollar su script de datos de usuario, consulte el GitHub repositorio de Amazon EC2 Auto Scaling.

Paso 3: Crear un grupo de Auto Scaling

Después de crear la plantilla de lanzamiento, cree un grupo de Auto Scaling.

Para crear un grupo de Auto Scaling

1. En la página Choose launch template or configuration (Elegir configuración o plantilla de lanzamiento), en Auto Scaling group name, ingrese un nombre para el grupo de Auto Scaling (**`TestAutoScalingEvent-group`**).
2. Elija Next (Siguiente) para ir a la página Choose instance launch options (Elegir opciones de lanzamiento de la instancia).
3. En Network (Red), elija una VPC.
4. En Availability Zones and subnets (Zonas de disponibilidad y subredes), elija una o varias subredes de una o varias zonas de disponibilidad.
5. En la sección Instance type requirements (Requisitos del tipo de instancia), utilice la configuración predeterminada para simplificar este paso. (No anule la plantilla de lanzamiento). En este tutorial, lanzará solo una On-Demand instancia con el tipo de instancia especificado en la plantilla de lanzamiento.
6. Elija Skip to review (Omitir para revisar) en la parte inferior de la pantalla.
7. En la página Review (Revisión), revise los detalles del grupo de Auto Scaling, y luego elija Create Auto Scaling group (Crear grupo de Auto Scaling).

Paso 4: agregar un enlace de ciclo de vida

Agregue un enlace de ciclo de vida para mantener la instancia en estado de espera hasta que se complete la acción de ciclo de vida.

Para agregar un enlace de ciclo de vida

1. Abra la página grupos de escalado automático en la consola de Amazon EC2.

2. Seleccione la casilla situada junto al grupo de escalado automático. Se abre un panel dividido en la parte inferior de la página.
3. En el panel inferior, en la pestaña Instance management (Administración de instancias), en Lifecycle hooks (Enlaces de ciclo de vida), elija Create lifecycle hook (Crear enlace de ciclo de vida).
4. Para definir un enlace de ciclo de vida para escalar horizontalmente (lanzamiento de instancias), haga lo siguiente:

   a. En Lifecycle hook name (Nombre de enlace de ciclo de vida), ingrese **`TestAutoScalingEvent-hook`**.

   b. En Lifecycle transition (Transición del ciclo de vida), elija Instance launch (Lanzamiento de instancia).

   c. En Heartbeat timeout (Tiempo de espera de latido), ingrese **`300`** como número de segundos que se debe esperar una devolución de llamada desde el script de datos de usuario.

   d. En Default result (Resultado predeterminado), elija ABANDON (Abandonar). Si el enlace agota el tiempo de espera sin recibir una devolución de llamada del script de datos de usuario, el grupo de Auto Scaling termina la nueva instancia.

   e. (Opcional) Mantenga Notification metadata (Metadatos de notificación) en blanco.
5. Seleccione Crear.

Paso 5: Probar y verificar la funcionalidad

Para probar la funcionalidad, actualice el grupo de Auto Scaling aumentando la capacidad deseada del grupo de Auto Scaling en 1. El script de datos de usuario se ejecuta y comienza a comprobar el estado de ciclo de vida de destino de la instancia poco después del lanzamiento de la instancia. El script cambia el nombre de host y envía una acción de devolución de llamada cuando el estado de ciclo de vida de destino es `InService`. Este proceso suele tardar solo unos segundos en finalizar.

Para aumentar el tamaño del grupo de Auto Scaling

1. Abra la página grupos de escalado automático en la consola de Amazon EC2.
2. Seleccione la casilla situada junto al grupo de escalado automático. Vea los detalles en un panel inferior mientras sigue viendo las filas superiores del panel superior.
3. En el panel inferior, en la pestaña Details (Detalles), elija Group details (Detalles de grupo), Edit (Editar).

4. En Desired capacity (Capacidad deseada), aumente el valor actual en 1.
5. Elija Actualizar. Mientras se está iniciando la instancia, la columna Status (Estado) del panel superior muestra el estado Updating capacity (Actualizando capacidad).

Después de aumentar la capacidad deseada, puede verificar que la instancia se ha lanzado correctamente y que no se ha terminado en la descripción de las actividades de escalado.

Para ver la actividad de escalado

1. Vuelva a la página Auto Scaling groups (Grupos de Auto Scaling) y seleccione su grupo.
2. En la pestaña Activity (Actividad), en Activity history (Historial de actividad), la columna Status (Estado) muestra si su grupo de Auto Scaling ha lanzado una instancia.
3. Si el script de datos de usuario falla una vez que ha transcurrido el periodo de tiempo de espera, verá una actividad de escalado con el estado `Canceled` y el mensaje de estado `Instance failed to complete user's Lifecycle Action: Lifecycle Action with token e85eb647-4fe0-4909-b341-a6c42EXAMPLE was abandoned: Lifecycle Action Completed with ABANDON Result`.

Paso 6: limpiar

Si ha terminado de trabajar con los recursos que ha creado para este tutorial, siga los pasos que figuran a continuación para eliminarlos.

Para eliminar el enlace de ciclo de vida

1. Abra la página grupos de escalado automático en la consola de Amazon EC2.
2. Seleccione la casilla situada junto al grupo de escalado automático.
3. En la pestaña Instance management (Administración de instancias), en Lifecycle hooks (Enlaces de ciclo de vida), elija el enlace de ciclo de vida (`TestAutoScalingEvent-hook`).
4. Elija Acciones, Eliminar.
5. Para confirmar, vuelva a elegir Delete.

Para eliminar la plantilla de lanzamiento

1. Abra la página Launch templates (Plantillas de lanzamiento) de la consola de Amazon EC2.

2. Seleccione la plantilla de lanzamiento (`TestAutoScalingEvent-template`) y elija Acciones, seguido de Eliminar plantilla.
3. Cuando se le pida la confirmación, escriba **`Delete`** para confirmar la eliminación de la plantilla de lanzamiento especificada y, a continuación, elija Delete (Eliminar).

Si ha terminado de trabajar con el grupo de Auto Scaling de ejemplo, elimínelo. También puede eliminar el rol de IAM y la política de permisos que ha creado.

Para eliminar el grupo de Auto Scaling

1. Abra la página grupos de escalado automático en la consola de Amazon EC2.
2. Seleccione la casilla de verificación situada junto al grupo de Auto Scaling (`TestAutoScalingEvent-group`) y elija Delete (Eliminar).
3. Cuando se le pida la confirmación, escriba **`delete`** para confirmar la eliminación del grupo de escalado automático especificado y, a continuación, elija Delete (Eliminar).

   Un icono de carga en la columna Name (Nombre) indica que el grupo de Auto Scaling se está eliminando. Terminar la instancia y eliminar el grupo tarda unos minutos.

Para eliminar el rol de IAM

1. Abra la página Roles en la consola de IAM.
2. Seleccione el rol de la función (`TestAutoScalingEvent-role`).
3. Elija Eliminar.
4. Cuando se le pida la confirmación, escriba el nombre del rol y, a continuación, elija Delete (Eliminar).

Para eliminar la política de IAM:

1. Abra la página de Policies (Políticas) de la consola de IAM.
2. Seleccione la política que creó (`TestAutoScalingEvent-policy`).
3. Elija Acciones, Eliminar.
4. Cuando se le pida la confirmación, escriba el nombre de la política y, a continuación, elija Delete (Eliminar).

Recursos relacionados

Los siguientes temas relacionados pueden resultarle útiles a la hora de desarrollar un código que invoque acciones en las instancias en función de los datos disponibles en los metadatos de la instancia.

- Recuperar el estado de ciclo de vida de destino a través de los metadatos de instancia. En esta sección, se describe el estado del ciclo de vida de otros casos de uso, como la terminación de una instancia.
- Adición de enlaces de ciclo de vida (consola). Este procedimiento muestra cómo agregar enlaces de ciclo de vida tanto para la escalar horizontalmente (lanzamiento de instancias) como para la reducir horizontalmente (instancias que finalizan o vuelven a un grupo caliente).
- Categorías de metadatos de instancia en la Guía del usuario de Amazon EC2. En este tema, se enumeran todas las categorías de metadatos de instancias que puede utilizar para invocar acciones en instancias de EC2.

Para ver un tutorial que muestra cómo usar Amazon EventBridge para crear reglas que invoquen funciones de Lambda en función de los eventos que ocurren en las instancias de su grupo de Auto Scaling, consulte. Tutorial: Configuración de un enlace de ciclo de vida que invoca una función Lambda

Tutorial: Configuración de un enlace de ciclo de vida que invoca una función Lambda

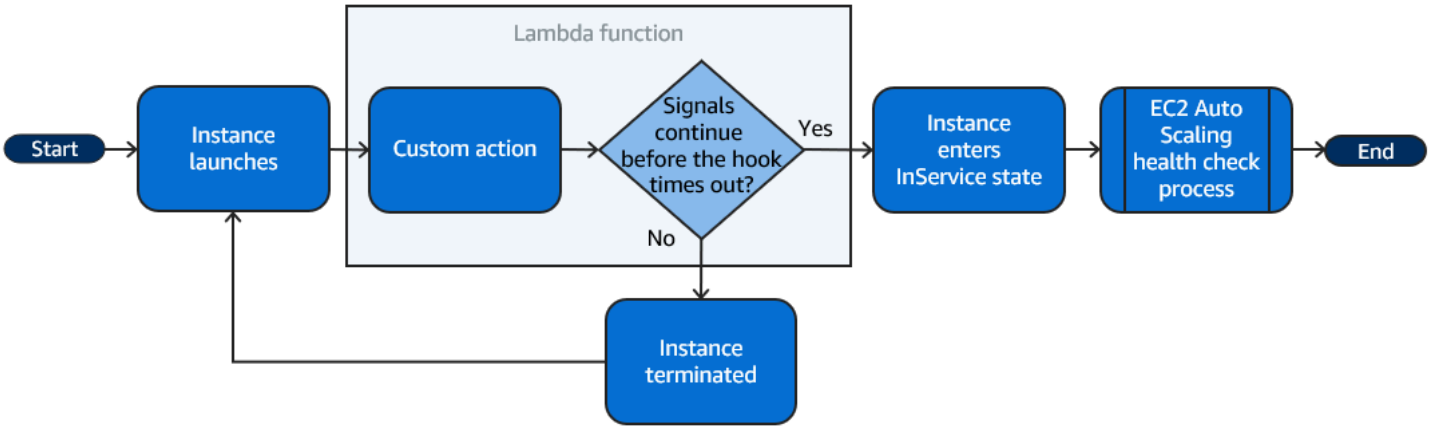
En este ejercicio, creará una EventBridge regla de Amazon que incluya un patrón de filtro que, si coincide, invoca una AWS Lambda función como objetivo de la regla. Proporcionamos el patrón de filtro y el código de la función de ejemplo que se van a usar.

Si todo está configurado correctamente, al final de este tutorial, la función Lambda realiza una acción personalizada cuando se inician las instancias. La acción personalizada simplemente registra el evento en la secuencia de CloudWatch registros asociada a la función Lambda.

La función Lambda también realiza una devolución de llamada para permitir que el ciclo de vida de la instancia continúe si esta acción se completa correctamente, pero permite que la instancia deje de iniciarse y termine si la acción falla.

En la siguiente ilustración se resume el flujo de un evento de escalado horizontal cuando utilice una función de Lambda para realizar una acción personalizada. Después de la inicialización de una

instancia, el ciclo de vida de la instancia se detiene hasta que se complete el enlace de ciclo de vida, ya sea porque se agota el tiempo de espera o porque Amazon EC2 Auto Scaling recibe una señal para continuar.



> **Note**
>
> Según tu caso de uso, puedes configurar un enlace de ciclo de vida siguiendo los pasos que se indican a continuación y creando una EventBridge regla. O bien, puedes usar una función de Lambda para configurar un enlace de ciclo de vida directamente sin crear una EventBridge regla.

Contenido

- Requisitos previos
- Paso 1: Crear un rol de IAM con permisos para completar acciones de ciclo de vida
- Paso 2: creación de una función de Lambda
- Paso 3: Crea una regla EventBridge
- Paso 4: agregar un enlace de ciclo de vida
- Paso 5: probar y verificar el evento
- Paso 6: limpiar
- Recursos relacionados

Requisitos previos

Antes de comenzar este tutorial, cree un grupo de Auto Scaling, si aún no dispone de uno. Para crear un grupo de escalado automático, abra la página Grupos de escalado automático en la consola de Amazon EC2 y elija Crear grupo de escalado automático.

Paso 1: Crear un rol de IAM con permisos para completar acciones de ciclo de vida

Antes de crear una función Lambda, primero debe crear un rol de ejecución y una política de permisos para permitir que Lambda complete los enlaces de ciclo de vida.

Para crear la política de

1. En la consola de IAM, abra la página Policies (Políticas), y, a continuación, elija Create policy (Crear política).
2. Seleccione la pestaña JSON.
3. En el cuadro Documento de política, pegue el siguiente documento de política en el cuadro y sustituya el texto por su número de cuenta y el nombre de su grupo de Auto Scaling. *italics*

   JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "autoscaling:CompleteLifecycleAction"
      ],
      "Resource":
 "arn:aws:autoscaling:*:123456789012:autoScalingGroup:*:autoScalingGroupName/
my-asg"
    }
  ]
}
```

4. Elija Siguiente.
5. Para Nombre de la política, introduzca **LogAutoScalingEvent-policy**. Elija Crear política.

Cuando termine de crear la política, podrá crear un rol que la utilice.

Para crear el rol.

1. En el panel de navegación de la izquierda, seleccione Roles.
2. Elija Crear rol.
3. En Select trusted entity (Seleccionar entidad de confianza), elija AWS service (Servicio de).
4. Para el caso de uso, elija Lambda y, luego, elija Next (Siguiente).
5. En Añadir permisos, elige la política que creaste (LogAutoScalingEvent-policy) y el nombre de la política AWSLambdaBasicExecutionRole. A continuación, elija Siguiente.

   > **Note**
   >
   > La AWSLambdaBasicExecutionRole política tiene los permisos que la función necesita para escribir CloudWatch registros en Logs.

6. En la página Name, review and create (Asignar nombre, revisar y crear), para Role name (Nombre del rol), ingrese **`LogAutoScalingEvent-role`** y seleccione Create role (Crear rol).

Paso 2: creación de una función de Lambda

Cree una función Lambda que actúe como destino de los eventos. La función Lambda de ejemplo, incluida Node.js, se invoca EventBridge cuando Amazon EC2 Auto Scaling emite un evento coincidente.

Cómo crear una función de Lambda

1. Abra la página Functions (Funciones) en la consola de Lambda.
2. Elija Create function (Crear función) y Author from scratch (Crear desde cero).
3. En Basic information (Información básica), para Function name (Nombre de función), escriba **`LogAutoScalingEvent`**.
4. Para Runtime, elija 18.x. Node.js
5. Elija Cambiar el rol de ejecución predeterminado y, a continuación, en Rol de ejecución, elija Usar un rol existente.
6. En Función existente, elija LogAutoScalingEvent-role.
7. Deje los demás valores predeterminados.
8. Seleccione Creación de función. Volverá al código y la configuración de la función.

9. Con la función `LogAutoScalingEvent` aún abierta en la consola, en Código fuente, en el editor, pegue el siguiente código de muestra en el archivo denominado index.mjs.

```
import { AutoScalingClient, CompleteLifecycleActionCommand } from "@aws-sdk/client-
auto-scaling";
export const handler = async(event) => {
  console.log('LogAutoScalingEvent');
  console.log('Received event:', JSON.stringify(event, null, 2));
  var autoscaling = new AutoScalingClient({ region: event.region });
  var eventDetail = event.detail;
  var params = {
    AutoScalingGroupName: eventDetail['AutoScalingGroupName'], /* required */
    LifecycleActionResult: 'CONTINUE', /* required */
    LifecycleHookName: eventDetail['LifecycleHookName'], /* required */
    InstanceId: eventDetail['EC2InstanceId'],
    LifecycleActionToken: eventDetail['LifecycleActionToken']
  };
  var response;
  const command = new CompleteLifecycleActionCommand(params);
  try {
    var data = await autoscaling.send(command);
    console.log(data); // successful response
    response = {
      statusCode: 200,
      body: JSON.stringify('SUCCESS'),
    };
  } catch (err) {
    console.log(err, err.stack); // an error occurred
    response = {
      statusCode: 500,
      body: JSON.stringify('ERROR'),
    };
  }
  return response;
};
```

Este código simplemente registra el evento para que, al final de este tutorial, pueda ver un evento en el flujo de registro de CloudWatch registros asociado a esta función de Lambda.

10. Elija Implementar.

Paso 3: Crea una regla EventBridge

Cree una EventBridge regla para ejecutar la función de Lambda. Para obtener más información acerca del uso de EventBridge, consulte Se usa EventBridge para gestionar eventos de Auto Scaling.

Para crear una regla con la consola

1. Abra la consola de EventBridge.
2. En el panel de navegación, seleccione Reglas.
3. Seleccione Creación de regla.
4. En Definir detalle de la regla, haga lo siguiente:
   a. En Nombre, escriba **`LogAutoScalingEvent-rule`**.
   b. En Bus de eventos, elija Predeterminado. Cuando un Servicio de AWS elemento de su cuenta genera un evento, siempre va al bus de eventos predeterminado de su cuenta.
   c. En Tipo de regla, elija Regla con un patrón de evento.
   d. Elija Siguiente.
5. En Crear patrón de evento, realice una de las siguientes acciones:
   a. En Fuente de eventos, elige AWS eventos o eventos EventBridge asociados.
   b. Desplácese hacia abajo hasta Patrón de eventos y haga lo siguiente:
   c. i. En Origen del evento, elija Servicios de AWS.
      ii. En Servicio de AWS, elija Auto Scaling.
      iii. En Event Type (Tipo de evento), seleccione Instance Launch and Terminate (Lanzamiento y terminación de la instancia).
      iv. De forma predeterminada, la regla coincide con cualquier evento de escalado o reducción horizontal. Para crear una regla que le notifique cuando se produzca un evento de escalamiento horizontal y una instancia pase a estar en estado de espera debido a un bloqueo del ciclo de vida, elija Eventos de instancia específicos y seleccione Instance-launch EC2 Lifecycle Action.
      v. De forma predeterminada, la regla coincide con cualquier grupo de Auto Scaling en la región. Para que la regla coincida con un grupo de escalado automático específico, elija Nombres de grupos específicos y, a continuación, seleccione el grupo.
      vi. Elija Siguiente.
6. En Seleccionar destino, realice una de las siguientes acciones:

a. Para Target types (Tipos de destino), elija Servicio de AWS.

b. En Seleccione destino, elija Función de Lambda.

c. En Function, elija. LogAutoScalingEvent

d. Seleccione Next (Siguiente) dos veces.

7. En la página Revisar y crear, elija Crear regla.

Paso 4: agregar un enlace de ciclo de vida

En esta sección, agrega un enlace de ciclo de vida para que Lambda ejecute la función en instancias en el momento del inicio.

Para agregar un enlace de ciclo de vida

1. Abra la página grupos de escalado automático en la consola de Amazon EC2.
2. Seleccione la casilla situada junto al grupo de escalado automático. Se abre un panel dividido en la parte inferior de la página.
3. En el panel inferior, en la pestaña Instance management (Administración de instancias), en Lifecycle hooks (Enlaces de ciclo de vida), elija Create lifecycle hook (Crear enlace de ciclo de vida).
4. Para definir un enlace de ciclo de vida para escalar horizontalmente (lanzamiento de instancias), haga lo siguiente:

   a. En Lifecycle hook name (Nombre de enlace de ciclo de vida), ingrese **`LogAutoScalingEvent-hook`**.

   b. En Lifecycle transition (Transición del ciclo de vida), elija Instance launch (Lanzamiento de instancia).

   c. En Heartbeat timeout (Tiempo de espera de latidos), ingrese **`300`** para indicar la cantidad de segundos que se debe esperar una devolución de llamada desde la función Lambda.

   d. En Default result (Resultado predeterminado), elija ABANDON (Abandonar). Esto significa que el grupo de Auto Scaling terminará una nueva instancia si el enlace agota el tiempo de espera sin recibir una devolución de llamada de la función Lambda.

   e. (Opcional) Deje la opción Notification metadata (Metadatos de notificación) vacía. Los datos del evento a los que pasamos EventBridge contienen toda la información necesaria para invocar la función Lambda.

5. Seleccione Crear.

Paso 5: probar y verificar el evento

Para probar el evento, actualice el grupo de Auto Scaling aumentando la capacidad deseada del grupo de Auto Scaling en 1. Se invoca la función Lambda pocos segundos después de aumentar la capacidad deseada.

Para aumentar el tamaño del grupo de Auto Scaling

1. Abra la página grupos de escalado automático en la consola de Amazon EC2.
2. Seleccione la casilla de verificación situada junto al grupo de Auto Scaling para ver los detalles en un panel inferior y seguir viendo las filas superiores del panel superior.
3. En el panel inferior, en la pestaña Details (Detalles), elija Group details (Detalles de grupo), Edit (Editar).
4. En Desired capacity (Capacidad deseada), aumente el valor actual en 1.
5. Elija Actualizar. Mientras se está iniciando la instancia, la columna Status (Estado) del panel superior muestra el estado Updating capacity (Actualizando capacidad).

Después de aumentar la capacidad deseada, puede verificar que se invocó la función Lambda.

Para ver la salida de la función Lambda

1. Abra la página de grupos de registros de la CloudWatch consola.
2. Seleccione el nombre del grupo de registros para la función Lambda (`/aws/lambda/LogAutoScalingEvent`).
3. Seleccione el nombre del flujo de registros para ver los datos proporcionados por la función para la acción del ciclo de vida.

A continuación, puede verificar que la instancia se haya iniciado correctamente a partir de la descripción de las actividades de escalado.

Para ver la actividad de escalado

1. Vuelva a la página Auto Scaling groups (Grupos de Auto Scaling) y seleccione su grupo.

2. En la pestaña Activity (Actividad), en Activity history (Historial de actividad), la columna Status (Estado) muestra si su grupo de Auto Scaling ha lanzado una instancia.

   - Si la acción se completó correctamente, la actividad de escalado tendrá el estado Successful (Correcto).
   - Si hubo un error, después de esperar unos minutos, verá una actividad de escalado con el estado Cancelled (Cancelado) y el mensaje: “Instance failed to complete user's Lifecycle Action: Lifecycle Action with token e85eb647-4fe0-4909-b341-a6c42EXAMPLE was abandoned: Lifecycle Action Completed with ABANDON Result” (La instancia no pudo completar la acción del ciclo de vida del usuario: se abandonó la acción del ciclo de vida con token e85eb647-4fe0-4909-b341-a6c42EJEMPLO: la acción del ciclo de vida se completó con el resultado ABANDON).

Para disminuir el tamaño del grupo de Auto Scaling

Si no necesita la instancia adicional que lanzó para esta prueba, puede abrir la pestaña Details (Detalles) y reducir el valor Desired capacity (Capacidad deseada) en 1.

Paso 6: limpiar

Si ha terminado de trabajar con los recursos que ha creado para este tutorial, siga los pasos a continuación para eliminarlos.

Para eliminar el enlace de ciclo de vida

1. Abra la página grupos de escalado automático en la consola de Amazon EC2.
2. Seleccione la casilla situada junto al grupo de escalado automático.
3. En la pestaña Instance management (Administración de instancias), en Lifecycle hooks (Enlaces de ciclo de vida), elija el enlace de ciclo de vida (`LogAutoScalingEvent-hook`).
4. Elija Acciones, Eliminar.
5. Para confirmar, vuelva a elegir Delete.

Para eliminar la EventBridge regla de Amazon

1. Abre la página de reglas en la EventBridge consola de Amazon.
2. En Event bus (Bus de eventos), elija el bus de eventos asociado a la regla (`Default`).
3. Active la casilla que hay junto a la regla (`LogAutoScalingEvent-rule`).

4. Elija Eliminar.
5. Cuando se le pida la confirmación, escriba el nombre de la regla y, a continuación, elija Delete (Eliminar).

Si ha terminado de trabajar con la función de ejemplo, elimínela. También puede eliminar el grupo de registro que almacena los registros de la función, y el rol de ejecución y la política de permisos que ha creado.

Para eliminar una función de Lambda

1. Abra la página Functions (Funciones) en la consola de Lambda.
2. Elija la función (`LogAutoScalingEvent`).
3. Elija Acciones, Eliminar.
4. Cuando se le pida la confirmación, escriba **`delete`** para confirmar la eliminación de la función especificada y, a continuación, elija Delete (Eliminar).

Para eliminar el grupo de registros

1. Abre la página de grupos de registro de la CloudWatch consola.
2. Seleccione el grupo de registros de la función (`/aws/lambda/LogAutoScalingEvent`).
3. Elija Acciones, Eliminar grupo(s) de registro(s).
4. En el cuadro de diálogo Eliminar grupos de registros, elija Eliminar.

Cómo eliminar el rol de ejecución

1. Abra la página Roles en la consola de IAM.
2. Seleccione el rol de la función (`LogAutoScalingEvent-role`).
3. Elija Eliminar.
4. Cuando se le pida la confirmación, escriba el nombre del rol y, a continuación, elija Delete (Eliminar).

Para eliminar la política de IAM:

1. Abra la página de Policies (Políticas) de la consola de IAM.

2. Seleccione la política que creó (`LogAutoScalingEvent-policy`).
3. Elija Acciones, Eliminar.
4. Cuando se le pida la confirmación, escriba el nombre de la política y, a continuación, elija Delete (Eliminar).

Recursos relacionados

Los siguientes temas relacionados pueden resultarte útiles a la hora de crear EventBridge reglas basadas en los eventos que ocurren en las instancias de tu grupo de Auto Scaling.

- Se usa EventBridge para gestionar eventos de Auto Scaling. En esta sección, se muestran ejemplos de eventos para otros casos de uso, incluidos los eventos para reducir horizontalmente.
- Adición de enlaces de ciclo de vida (consola). Este procedimiento muestra cómo agregar enlaces de ciclo de vida tanto para la escalar horizontalmente (lanzamiento de instancias) como para la reducir horizontalmente (instancias que finalizan o vuelven a un grupo caliente).

Para ver un tutorial que muestra cómo usar el Servicio de metadatos de instancias (IMDS) para invocar una acción desde la propia instancia, consulte Tutorial: Uso del script de datos y los metadatos de la instancia para recuperar el estado del ciclo de vida.

Reducción de la latencia de las aplicaciones con tiempos de arranque prolongados mediante grupos de calentamiento

Un grupo de calentamiento le permite reducir la latencia de las aplicaciones que tienen tiempos de arranque excepcionalmente prolongados, por ejemplo, porque las instancias tienen que escribir cantidades masivas de datos en el disco. Con los grupos de calentamiento, ya no tiene que aprovisionar en exceso los grupos de Auto Scaling para administrar la latencia con el fin de mejorar el rendimiento de las aplicaciones. Para obtener más información, consulte la siguiente entrada del blog Scaling your applications faster with EC2 Auto Scaling Warm Pools.

> **Important**
>
> Crear un grupo de calentamiento cuando no es necesario puede generar costos innecesarios. Si el primer tiempo de arranque de la aplicación no causa problemas de latencia detectables, probablemente no sea necesario que use un grupo de calentamiento.

Temas

- Conceptos clave
- Requisitos previos
- Actualización de las instancias de un grupo en caliente
- Recursos relacionados
- Limitaciones
- Uso de enlaces de ciclo de vida con un grupo de calentamiento en un grupo de escalado automático
- Creación de un grupo en caliente para un grupo de escalado automático
- Visualización del estado de la comprobación de estado y el motivo de los errores de la comprobación de estado
- Ejemplos de cómo crear y gestionar piscinas termales con el AWS CLI

Conceptos clave

Antes de empezar, familiarícese con los siguientes conceptos clave:

Grupo de calentamiento

Un grupo de calentamiento es un grupo de instancias de EC2 inicializadas previamente que se encuentra junto a un grupo de escalado automático. Siempre que la aplicación tenga que escalarse horizontalmente, el grupo de escalado automático puede recurrir al grupo de calentamiento para satisfacer su nueva capacidad deseada. Esto ayuda a garantizar que las instancias estén listas para comenzar a atender rápidamente el tráfico de las aplicaciones, lo que acelera la respuesta a un evento de escalado horizontal. A medida que las instancias salgan del grupo de calentamiento, cuentan para alcanzar la capacidad deseada del grupo. Esto se conoce como arranque en caliente.

Mientras las instancias se encuentran en el grupo de preparación, las políticas de escalado solo escalan horizontalmente si el valor de la métrica de las instancias con estado `InService` es mayor que el umbral superior de la alarma de la política de escalado (que es el mismo que la utilización de objetivo de una política de escalado de seguimiento de objetivo).

Tamaño del grupo de calentamiento

De forma predeterminada, el tamaño del grupo de calentamiento se calcula como la diferencia entre la capacidad máxima del grupo de escalado automático y la capacidad deseada. Por

ejemplo, si la capacidad deseada de su grupo de escalado automático es 6, y la capacidad máxima es 10, el tamaño del grupo de calentamiento será 4 cuando configure el grupo de calentamiento por primera vez y el grupo se inicialice.

Para especificar la capacidad máxima del grupo de calentamiento por separado, utilice la opción de especificación personalizada (`MaxGroupPreparedCapacity`) y establezca un valor personalizado que sea mayor que la capacidad actual del grupo. Si especifica un valor personalizado, el tamaño del grupo de calentamiento se calcula como la diferencia entre el valor personalizado y la capacidad actual deseada del grupo. Por ejemplo, si la capacidad deseada de su grupo de escalado automático es 6, si la capacidad máxima es 20, y si el valor personalizado es 8, el tamaño del grupo de calentamiento será 2 cuando configure el grupo de calentamiento por primera vez y el grupo se inicialice.

Es posible que solo necesite usar la especificación personalizada (`MaxGroupPreparedCapacity`) cuando trabaje con grupos de escalado automático grandes para administrar la rentabilidad de tener un grupo de calentamiento. Por ejemplo, un grupo de escalado automático con 1000 instancias, una capacidad máxima de 1500 (para proporcionar una capacidad adicional para picos de tráfico de emergencia), y un grupo de calentamiento de 100 instancias podría servirle para alcanzar sus objetivos mejor que mantener 500 instancias reservadas para el uso futuro dentro del grupo de calentamiento.

Tamaño mínimo de grupo de calentamiento

Considere la opción de usar la configuración de tamaño mínimo (`MinSize`) para establecer de forma estática el número mínimo de instancias que se mantendrán en el grupo en caliente. No hay un tamaño mínimo establecido de manera predeterminada. La configuración `MinSize` es útil cuando se especifica `MaxGroupPreparedCapacity` para garantizar que se mantenga una cantidad mínima de instancias en el grupo en caliente, incluso cuando la capacidad deseada del grupo de escalado automático sea mayor que `MaxGroupPreparedCapacity`.

Estado de la instancia del grupo de calentamiento

Puede mantener las instancias en el grupo de calentamiento en uno de tres estados: `Stopped`, `Running` o `Hibernated`. Mantener las instancias en un estado `Stopped` es una manera efectiva de minimizar los costos. Con las instancias detenidas, solo paga por los volúmenes que utiliza y las direcciones IP elásticas adjuntas a las instancias.

También puede mantener las instancias en un estado `Hibernated` para detener instancias sin eliminar el contenido de la memoria (RAM). Cuando se hiberna una instancia, esto indica al sistema operativo que guarde el contenido de la RAM en el volumen raíz de Amazon EBS.

Cuando se reinicia la instancia, el volumen raíz se restaura a su estado anterior y el contenido de la RAM se vuelve a cargar. Mientras las instancias están en hibernación, solo paga por los volúmenes de EBS, incluido el almacenamiento del contenido de la RAM y las direcciones IP elásticas adjuntas a las instancias.

Mantener las instancias en un estado `Running` dentro del grupo de calentamiento también es posible, pero se desaconseja encarecidamente evitar incurrir en cargos innecesarios. Cuando las instancias se detienen o hibernan, ahorra el costo de las mismas instancias. Solo paga por las instancias cuando se están ejecutando.

Enlaces de ciclo de vida

Los enlaces de ciclo de vida se usan para poner instancias en estado de espera a fin de poder realizar acciones personalizadas en las instancias. Las acciones personalizadas se realizan a medida que se lanzan las instancias o antes de que finalicen.

En la configuración de un grupo en caliente, los enlaces de ciclo de vida retrasan la detención o la hibernación de las instancias, así como su puesta en servicio, durante un evento de escalado horizontal hasta que hayan terminado de inicializarse. Si agrega un grupo de calentamiento al grupo de escalado automático sin un enlace de ciclo de vida, las instancias que tardan mucho tiempo en finalizar la inicialización se podrían detener o hibernar y, a continuación, poner en servicio durante un evento de escalado horizontal antes de que estén listas.

Política de reutilización de instancias

De forma predeterminada, Amazon EC2 Auto Scaling termina las instancias cuando se escala el grupo de escalado automático. A continuación, lanza nuevas instancias en el grupo de calentamiento para reemplazar las instancias que se terminaron.

Si desea devolver instancias al grupo de calentamiento, puede especificar una política de reutilización de instancias. Esto le permite reutilizar instancias que ya están configuradas para atender el tráfico de aplicaciones. Para asegurarse de que el grupo de calentamiento no esté sobreaprovisionado, Amazon EC2 Auto Scaling puede terminar instancias del grupo de calentamiento para reducir su tamaño cuando es mayor de lo necesario en función de su configuración. Al terminar instancias en el grupo de calentamiento, utiliza la política de terminación predeterminada para elegir qué instancias terminará primero.

> **Important**
>
> Si desea hibernar instancias durante la reducción horizontal y hay instancias existentes en el grupo de escalado automático, deben cumplir los requisitos de hibernación de

instancias. Si no lo hacen, cuando las instancias regresen al grupo de calentamiento, volverán a detenerse en lugar de hibernar.

> **Note**
>
> Actualmente, solo puede especificar una política de reutilización de instancias mediante la AWS CLI o un SDK. Esta característica no está disponible desde la consola.

Requisitos previos

Antes de crear un grupo en caliente para su grupo de escalado automático, decida cómo utilizará los enlaces de ciclo de vida para inicializar nuevas instancias con un estado inicial adecuado.

Para realizar acciones personalizadas en las instancias mientras están en estado de espera debido a un enlace de ciclo de vida, tiene dos opciones:

- Para escenarios sencillos en los que desea ejecutar comandos en las instancias durante el lanzamiento, puede incluir un script de datos de usuario al crear una plantilla de lanzamiento o una configuración de lanzamiento para el grupo de escalado automático. Los scripts de datos de usuario son solo scripts de intérprete de comandos normales o directivas cloud-init que ejecutan cloud-init cuando se inician las instancias. El script también puede controlar cuándo las instancias realizan la transición al siguiente estado utilizando el ID de la instancia en la que se ejecuta. Si aún no lo ha hecho, actualice el script para recuperar el ID de instancia de la instancia de los metadatos de instancia. Para obtener más información, consulte Metadatos de instancia de acceso en la Guía del usuario de Amazon EC2.

> **Tip**
>
> Para ejecutar scripts de datos de usuario cuando se reinicia una instancia, estos datos deben tener el formato multiparte MIME y especificar lo siguiente en la sección `#cloud-config` de los datos de usuario:
>
> ```
> #cloud-config
> cloud_final_modules:
>  - [scripts-user, always]
> ```

- En situaciones avanzadas en las que necesitas un servicio, como AWS Lambda hacer algo cuando las instancias entran o salen del grupo activo, puedes crear un enlace de ciclo de vida para tu grupo de Auto Scaling y configurar el servicio de destino para que realice acciones personalizadas en función de las notificaciones del ciclo de vida. Para obtener más información, consulte Destinos de notificación admitidos.

Preparación de instancias para la hibernación

Para preparar instancias de escalado automático para utilizar el estado de grupo `Hibernated`, cree una nueva plantilla o una configuración de inicialización que esté configurada correctamente para admitir la hibernación de instancias, como se describe en el tema Requisitos previos de la hibernación de la Guía del usuario de Amazon EC2. A continuación, asocie la nueva plantilla de lanzamiento o configuración de lanzamiento con el grupo de escalado automático e inicie una actualización de instancias para reemplazar las instancias asociadas con una plantilla de lanzamiento o configuración de lanzamiento anterior. Para obtener más información, consulte Uso de una actualización de instancias para actualizar instancias en un grupo de escalado automático.

Actualización de las instancias de un grupo en caliente

Para actualizar las instancias de un grupo en caliente, cree una nueva plantilla de lanzamiento o configuración de lanzamiento y asóciela al grupo de escalado automático. Las nuevas instancias se lanzan con la nueva AMI y otras actualizaciones especificadas en la plantilla de lanzamiento o configuración de lanzamiento, pero las instancias existentes no resultan afectadas.

Puede iniciar una actualización de instancias para hacer una actualización de su grupo a fin de forzar el lanzamiento de instancias de reemplazo de grupo en caliente que utilicen la nueva plantilla de lanzamiento o configuración de lanzamiento. Una actualización de instancia reemplaza primero las instancias `InService`. Luego reemplaza las instancias en el grupo de calentamiento. Para obtener más información, consulte Uso de una actualización de instancias para actualizar instancias en un grupo de escalado automático.

Recursos relacionados

Puedes visitar nuestro GitHub repositorio para ver ejemplos de enlaces de ciclo de vida para piscinas templadas.

Limitaciones

- Limitaciones de los grupos en caliente para un grupo de escalado automático con tipos de instancias mixtos:
  - Los grupos en caliente no son compatibles con los grupos de instancias mixtos ponderados. Si su grupo de escalado automático usa la ponderación de instancias, no puede agregar un grupo en caliente.
  - Los grupos en caliente no son compatibles con las instancias de spot en grupos de instancias mixtos. Tu política de instancias mixtas debe configurarse solo para On-Demand las instancias que usen grupos inactivos.
  - Si utiliza grupos en caliente con grupos de instancias mixtos en estado de hibernación, debe configurar `HibernationOptions` en su plantilla de lanzamiento.
- Amazon EC2 Auto Scaling puede poner una instancia en un estado `Stopped` o `Hibernated` solo si tiene un volumen de Amazon EBS como dispositivo raíz. Las instancias que utilizan almacenes de instancias para el dispositivo raíz no se pueden detener o hibernar.
- Amazon EC2 Auto Scaling puede colocar una instancia en un estado `Hibernated` solo si cumple todos los requisitos enumerados en el tema Requisitos previos de la hibernación de la Guía del usuario de Amazon EC2.
- Si el grupo de calentamiento se agota cuando hay un evento de escalado horizontal, las instancias se iniciarán directamente en el grupo de escalado automático (un arranque en frío). También puede experimentar arranques en frío si no queda capacidad en una zona de disponibilidad.
- Si una instancia del grupo de calentamiento encuentra un problema durante el proceso de inicialización, que le impide que alcance el estado `InService`, la instancia se considerará una inicialización fallida y se cancelará. Esto se aplica independientemente de la causa subyacente, como un error de capacidad insuficiente o cualquier otro factor.
- Amazon EC2 Auto Scaling detiene o hiberna las instancias a medida que ingresan al grupo activo y no espera a que los datos del usuario terminen de ejecutarse. Como resultado, la acción de detener o hibernar puede interrumpir los datos de usuario que han estado funcionando durante mucho tiempo. Para obtener más información, consulte Uso de enlaces de ciclo de vida con un grupo de calentamiento en un grupo de escalado automático.
- Si intenta utilizar un grupo de calentamiento con un grupo de nodos administrados de Amazon Elastic Kubernetes Service (Amazon EKS), puede que las instancias que aún estén inicializándose se registren con el clúster de Amazon EKS. Como resultado, puede que el clúster programe trabajos en una instancia que se está preparando para detenerse o hibernarse.

- Del mismo modo, si intenta utilizar un grupo de calentamiento con un clúster de Amazon ECS, las instancias pueden registrarse en el clúster antes de que terminen de inicializarse. Para resolver este problema, debe configurar una plantilla de lanzamiento o una configuración de lanzamiento que incluya una variable de configuración de agente especial en los datos de usuario. Para obtener más información, consulte Uso de un grupo de calentamiento para el grupo de escalado automático en la Guía para desarrolladores de Amazon Elastic Container Service.

Uso de enlaces de ciclo de vida con un grupo de calentamiento en un grupo de escalado automático

Las instancias del grupo de calentamiento mantienen su propio ciclo de vida independiente para ayudarlo a crear las acciones personalizadas adecuadas para cada transición. Este ciclo de vida está diseñado para ayudarlo a invocar acciones en un servicio de destino (por ejemplo, una función Lambda) mientras aún se está inicializando una instancia y antes de ponerla en servicio.

> **Note**
>
> Las operaciones de la API que utiliza para agregar y administrar enlaces de ciclo de vida y completar acciones de ciclo de vida no se modifican. Solo se modifica el ciclo de vida de la instancia.

Para obtener más información acerca de cómo agregar un enlace de ciclo de vida, consulte Incorporación de los enlaces de ciclo de vida en su grupo de escalado automático. Para obtener más información sobre cómo completar una acción de ciclo de vida, consulte Cómo completar una acción del ciclo de vida en un grupo de escalado automático.

Es posible que necesite un enlace de ciclo de vida para las instancias que entran en el grupo de calentamiento por uno de los siguientes motivos:

- Desea lanzar instancias de EC2 desde una AMI que tarda mucho tiempo en finalizar la inicialización.
- Desea ejecutar scripts de datos de usuario para arrancar las instancias de EC2.

Important

Tan pronto como una instancia termina de lanzarse a la piscina caliente, Amazon EC2 Auto Scaling detiene la instancia o la hiberna. Amazon EC2 Auto Scaling no espera a que los datos del usuario terminen de ejecutarse, por lo que la acción de detener o hibernar puede interrumpir un script que aún se esté ejecutando.
De forma predeterminada, cloud-init ejecuta los datos del usuario solo en el primer arranque. Un script interrumpido no se vuelve a ejecutar cuando la instancia sale del grupo activo y entra en servicio. La instancia puede alcanzar el `InService` estado sin terminar su script de arranque y Amazon EC2 Auto Scaling informa que la instancia está en buen estado.
Para evitar un arranque incompleto, utilice una de las siguientes opciones:

- Añada un enlace de ciclo de vida para la acción del `EC2_INSTANCE_LAUNCHING` ciclo de vida. A continuación, la instancia permanece en ese `Warmed:Pending:Wait` estado hasta que el script de arranque finalice y llame`CompleteLifecycleAction`.
- Incluya los pasos de arranque en su AMI para que quede poco trabajo o nada de trabajo para los datos del usuario.
- Configure los datos de usuario para que se ejecuten en cada arranque, tal y como se describe enRequisitos previos. A continuación, el script se vuelve a ejecutar cuando la instancia se reinicia al salir de la piscina caliente.

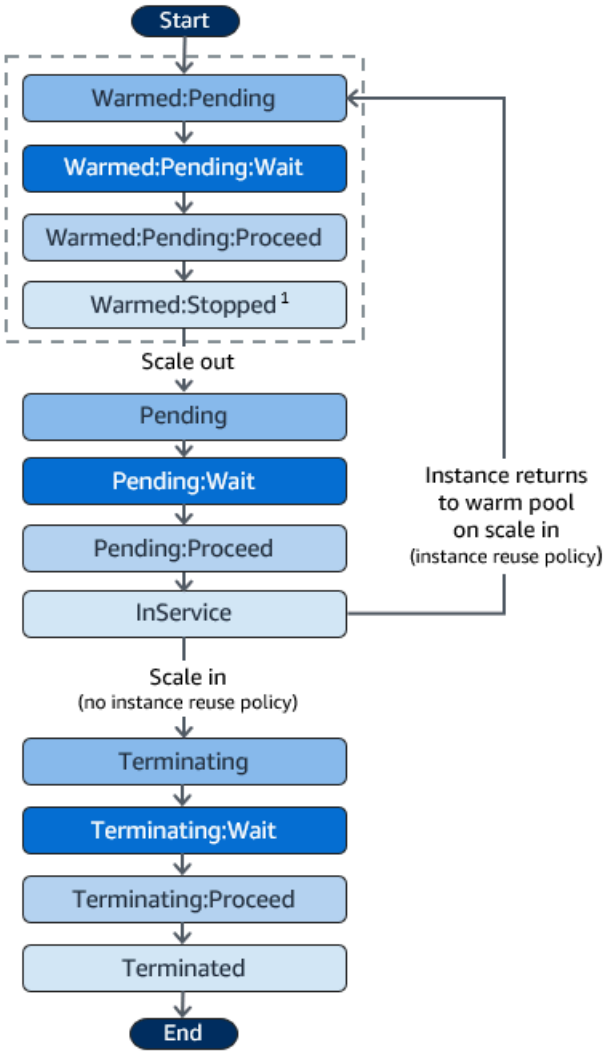
Es posible que necesite un enlace de ciclo de vida para las instancias que salen del grupo de calentamiento por uno de los siguientes motivos:

- Puede utilizar más tiempo para preparar instancias de EC2 para su uso. Por ejemplo, para que la aplicación pueda funcionar correctamente, es posible que tenga servicios que deban iniciarse cuando se reinicia una instancia.
- Desea rellenar previamente los datos de caché para que un nuevo servidor no se lance con una caché vacía.
- Desea registrar nuevas instancias como instancias administradas con su servicio de administración de configuraciones.

Transiciones de estado del ciclo de vida para las instancias de un grupo de calentamiento

Como parte de su ciclo de vida, una instancia de Auto Scaling puede pasar por muchos estados.

En el diagrama a continuación, se muestra la transición entre los estados de Auto Scaling cuando utiliza un grupo de calentamiento:



[1] Este estado varía según la configuración del estado del grupo de calentamiento. Si el estado del grupo se configura en `Running`, este estado es `Warmed:Running` en su lugar. Si el estado del grupo se configura en `Hibernated`, este estado es `Warmed:Hibernated` en su lugar.

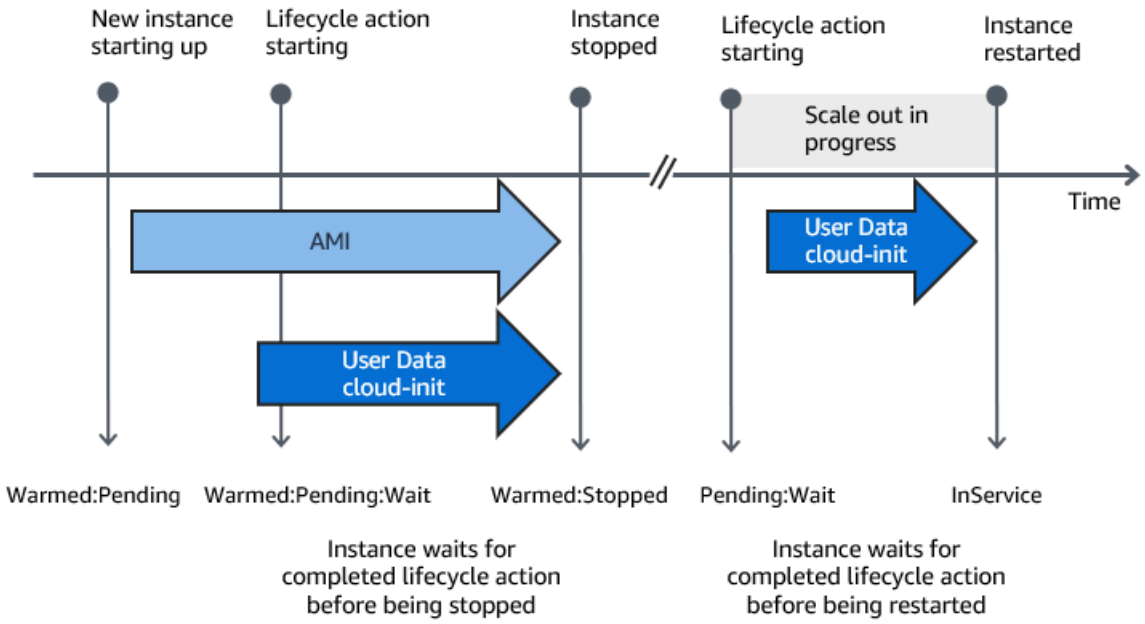
Cuando agregue enlaces de ciclo de vida, tenga en cuenta lo siguiente:

- Cuando se configura un enlace de ciclo de vida para la acción de ciclo de vida `autoscaling:EC2_INSTANCE_LAUNCHING`, una instancia recién lanzada se detiene primero

para realizar una acción personalizada cuando alcanza el estado `Warmed:Pending:Wait`, y nuevamente cuando la instancia se reinicia y alcanza el estado `Pending:Wait`.

- Cuando se configura un enlace de ciclo de vida para la acción de ciclo de vida `EC2_INSTANCE_TERMINATING`, una instancia que termina hace una pausa para realizar una acción personalizada cuando alcanza el estado `Terminating:Wait`. Sin embargo, si especifica una política de reutilización de instancias para devolver instancias al grupo de calentamiento en reducción horizontal en lugar de terminarlas, una instancia que vuelve al grupo de calentamiento se pausa para llevar a cabo una acción personalizada en el estado `Warmed:Pending:Wait` para la acción del ciclo de vida `EC2_INSTANCE_TERMINATING`.
- Si la demanda de la aplicación agota el grupo de calentamiento, Amazon EC2 Auto Scaling puede lanzar instancias directamente en el grupo de escalado automático siempre y cuando el grupo aún no haya alcanzado su capacidad máxima. Si las instancias se lanzan directamente en el grupo, solo se detendrán para realizar una acción personalizada en el estado `Pending:Wait`.
- Para controlar cuánto tiempo permanece una instancia en un estado de espera antes de pasar al siguiente estado, configure la acción personalizada para usar el comando complete-lifecycle-action. Con los enlaces del ciclo de vida, las instancias se mantienen en estado de espera hasta que se notifica a Amazon EC2 Auto Scaling que la acción del ciclo de vida se ha completado o hasta que finaliza el tiempo de espera (que, de forma predeterminada, es de una hora).

A continuación, se resume el flujo de un evento de escalado horizontal.



Cuando las instancias alcanzan un estado de espera, Amazon EC2 Auto Scaling envía una notificación. Hay ejemplos de estas notificaciones disponibles en la EventBridge sección de esta guía. Para obtener más información, consulte Ejemplos de eventos y patrones de grupos en caliente.

Destinos de notificación admitidos

Amazon EC2 Auto Scaling proporciona soporte para definir cualquiera de los siguientes como destinos de notificación para notificaciones de ciclo de vida:

- EventBridge reglas
- Temas de Amazon SNS
- Colas de Amazon SQS
- AWS Lambda funciones

> **Important**
>
> Si tiene un script de datos de usuario (cloud-init) en la plantilla de lanzamiento o en la configuración de lanzamiento que configura las instancias cuando se lanzan, no es necesario que reciba notificaciones para llevar a cabo acciones personalizadas en las instancias que se están iniciando o reiniciando.

Las secciones que se indican a continuación contienen vínculos a la documentación que describe cómo configurar los destinos de notificación:

EventBridge reglas: para ejecutar código cuando Amazon EC2 Auto Scaling pone una instancia en estado de espera, puede crear una EventBridge regla y especificar una función de Lambda como objetivo. Para invocar distintas funciones Lambda basadas en distintas notificaciones de ciclo de vida, puede crear varias reglas y asociar cada regla con un patrón de evento específico y una función Lambda. Para obtener más información, consulte Crea EventBridge reglas para los eventos de piscina caliente.

Temas de Amazon SNS: para recibir una notificación cuando una instancia se coloca en estado de espera, puede crear un tema de Amazon SNS y, a continuación, configurar el filtro de mensajes de Amazon SNS para entregar notificaciones de ciclo de vida de forma diferente según un atributo de mensaje. Para obtener más información, consulte Recepción de notificaciones mediante Amazon SNS.

Colas de Amazon SQS: para configurar un punto de entrega para las notificaciones de ciclo de vida en el que un consumidor relevante pueda recogerlas y procesarlas, puede crear una cola de Amazon SQS y un consumidor de colas que procese los mensajes de la cola de SQS. Si desea que el consumidor de la cola procese las notificaciones de ciclo de vida de forma diferente en función de un atributo de mensaje, también debe configurar el consumidor de la cola para analizar el mensaje y, a continuación, actuar sobre el mensaje cuando un atributo específico coincide con el valor deseado. Para obtener más información, consulte Recepción de notificaciones mediante Amazon SQS.

AWS Lambda funciones: para ejecutar código personalizado cuando Amazon EC2 Auto Scaling pone una instancia en estado de espera, puede especificar una función de Lambda como objetivo de notificación. La función Lambda se invoca con los datos de notificación del ciclo de vida, lo que le permite realizar acciones personalizadas, como la configuración de instancias, la configuración de aplicaciones o la integración con otros servicios. AWS Debe configurar la política basada en recursos de la función de Lambda para permitir al rol vinculado al servicio de Auto Scaling que invoque la función. Para obtener más información, consulte Dirige las notificaciones a AWS Lambda directamente.

Creación de un grupo en caliente para un grupo de escalado automático

En este tema, se describe cómo crear un grupo en caliente para su grupo de escalado automático.

> **Important**
>
> Antes de continuar, complete los requisitos previos para crear un grupo en caliente y confirme que ha creado un enlace de ciclo de vida para su grupo de escalado automático.

Creación de un grupo de calentamiento

Utilice el siguiente procedimiento a fin de crear un grupo en caliente para el grupo de escalado automático.

Para crear un grupo de calentamiento (consola)

1. Abra la consola de Amazon EC2 en https://console.aws.amazon.com/ec2/ y seleccione Auto Scaling Groups en el panel de navegación.
2. Seleccione la casilla de verificación situada junto a un grupo existente.

   Se abre un panel dividido en la parte inferior de la página.

3. Elija la pestaña Instance management (Administración de instancias).
4. En Warm pool (Grupo de calentamiento), elija Create warm pool (Crear grupo de calentamiento).
5. Para configurar un grupo de calentamiento, haga lo siguiente:

   a. EnWarm pool instance state (Estado de la instancia del grupo de calentamiento), elija el estado al que desea mover las instancias cuando entren al grupo de calentamiento. El valor predeterminado es `Stopped`.

   b. En Minimum warm pool size (Tamaño mínimo de grupo de calentamiento), ingrese el número mínimo de instancias que se van a mantener en el grupo de calentamiento.

   c. Para la reutilización de instancias, seleccione la casilla Reutilización el reducir horizontalmente para permitir que las instancias del grupo de escalado automático vuelvan al grupo de calentamiento que se redujo horizontalmente.

   d. En Tamaño del grupo de calentamiento, elija una de las opciones disponibles:

      - Especificación predeterminada: el tamaño del grupo de calentamiento está determinado por la diferencia entre la capacidad máxima y deseada del grupo de escalado automático. Esta opción optimiza la administración del grupo de calentamiento. Después de crear el grupo de calentamiento, su tamaño se puede actualizar fácilmente con solo ajustar la capacidad máxima del grupo.
      - Especificación personalizada: el tamaño del grupo de calentamiento está determinado por la diferencia entre un valor personalizado y la capacidad deseada del grupo de escalado automático. Esta opción le brinda flexibilidad para administrar el tamaño de su grupo de calentamiento independientemente de la capacidad máxima del grupo.

6. Consulte la sección Tamaño estimado del grupo de calentamiento en función de la configuración actual para confirmar cómo se aplica la especificación predeterminada o personalizada al tamaño del grupo de calentamiento. Recuerde que el tamaño del grupo de calentamiento depende de la capacidad deseada del grupo de escalado automático, que cambiará si el grupo escala.
7. Seleccione Crear.

Selección del tipo de instancia con grupos de instancias mixtos

Auto Scaling prioriza los tipos de instancias que ya están en el grupo en caliente durante los eventos de escalado cuando su grupo está configurado con una política de instancias mixtas. Comportamiento de lanzamiento:

1. Auto Scaling intenta lanzar instancias mediante los tipos de instancia disponibles en el grupo en caliente.
2. Si el inicio en caliente falla, Auto Scaling intentará el lanzamiento en frío utilizando todos los tipos de instancias restantes de la política de instancias mixtas.

Example

Ejemplo

Configure su grupo de escalado automático con 10 tipos de instancias, mientras que su grupo en caliente contiene 6 de esos tipos de instancias. Durante el escalado horizontal, Auto Scaling primero prueba los 6 tipos de instancias del grupo en caliente. Si no funciona, Auto Scaling prueba todos los tipos de instancias configurados mediante un lanzamiento en frío.

De esta forma, podrá aprovechar al máximo el rendimiento cuando sea posible y, al mismo tiempo, mantener la flexibilidad de su configuración completa de instancias mixtas.

Eliminación de un grupo de calentamiento

Cuando ya no necesite el grupo de calentamiento, utilice el siguiente procedimiento para eliminarlo.

Para eliminar el grupo de calentamiento (consola)

1. Abra la consola de Amazon EC2 en https://console.aws.amazon.com/ec2/ y seleccione Auto Scaling Groups en el panel de navegación.
2. Seleccione la casilla de verificación situada junto a un grupo existente.

   Se abre un panel dividido en la parte inferior de la página.
3. Elija la pestaña Instance management (Administración de instancias).
4. En Warm pool (Grupo de preparación), elija Actions (Acciones), Delete (Eliminar).
5. Cuando se le pida confirmación, seleccione Eliminar.

Visualización del estado de la comprobación de estado y el motivo de los errores de la comprobación de estado

Las comprobaciones de estado permiten que Amazon EC2 Auto Scaling determine cuándo una instancia no tiene un estado correcto y debe terminarse. Para instancias de un grupo de

calentamiento que se mantienen en un estado `Stopped`, emplea la información que Amazon EBS tiene de la disponibilidad de una instancia `Stopped` para identificar las instancias con un estado incorrecto. Para esto, llama a la API `DescribeVolumeStatus` para determinar el estado del volumen de EBS asociado a la instancia. Para las instancias de un grupo de calentamiento que se mantienen en un estado `Running`, se basa en las comprobaciones de estado de EC2 para determinar el estado de la instancia. Aunque no existe un periodo de gracia de comprobación de estado de las instancias de un grupo de calentamiento, Amazon EC2 Auto Scaling no comienza a comprobar el estado de la instancia hasta que finalice el enlace de ciclo de vida.

Cuando se comprueba que una instancia tiene un estado incorrecto, Amazon EC2 Auto Scaling elimina automáticamente la instancia en mal estado y crea una nueva para reemplazarla. Por lo general, las instancias se terminan unos minutos después de no superar la comprobación de estado. Para obtener más información, consulte Visionado del motivo de los errores de una comprobación de estado.

También se admiten comprobaciones de estado personalizadas. Esto puede resultar útil si tiene su propio sistema de comprobación de estado que pueda detectar el estado de una instancia y enviar esta información a Amazon EC2 Auto Scaling. Para obtener más información, consulte Configuración de una comprobación de estado personalizada para el grupo de escalado automático.

En la consola de Amazon EC2 Auto Scaling puede ver el estado (correcto o incorrecto) de las instancias de un grupo de calentamiento. También puede ver su estado de salud mediante uno de los SDK AWS CLI o uno de ellos.

Para ver el estado de las instancias de un grupo de calentamiento (consola)

1. Abra la consola de Amazon EC2 en https://console.aws.amazon.com/ec2/ y seleccione Auto Scaling Groups en el panel de navegación.
2. Seleccione la casilla situada junto al grupo de escalado automático.

   Se abre un panel dividido en la parte inferior de la página Auto Scaling groups (Grupos de Auto Scaling).
3. En la pestaña Instance management (Administración de instancia), en Warm pool instances (Instancias de grupo de calentamiento), la columna Lifecycle (Ciclo de vida) muestra el estado de las instancias.

   La columna Health status (Estado) muestra la evaluación que Amazon EC2 Auto Scaling ha realizado sobre el estado de la instancia.

> **Note**
>
> El estado de las nuevas instancias es correcto. Hasta que no finalice el enlace de ciclo de vida, no se comprueba el estado de una instancia.

Para ver el motivo de los errores de una comprobación de estado (consola)

1. Abra la consola de Amazon EC2 en https://console.aws.amazon.com/ec2/ y seleccione Auto Scaling Groups en el panel de navegación.
2. Seleccione la casilla situada junto al grupo de escalado automático.

   Se abre un panel dividido en la parte inferior de la página Auto Scaling groups (Grupos de Auto Scaling).
3. En la pestaña Activity (Actividad), en Activity history (Historial de actividad), la columna Status (Estado) muestra si su grupo de escalado automático ha lanzado o terminado las instancias correctamente.

   Si terminó cualquier instancia en mal estado, la columna Cause (Causa) muestra la fecha y la hora de la terminación y el motivo del error de la comprobación de estado. Por ejemplo, «El 1 de abril de 2021, T21:48:35Z una instancia quedó fuera de servicio debido a un error en la comprobación del estado del volumen de EBS».

Para ver el estado de las instancias de su piscina climatizada (AWS CLI)

Para ver el grupo de calentamiento de un grupo de escalado automático, utilice el siguiente comando describe-warm-pool.

```
aws autoscaling describe-warm-pool --auto-scaling-group-name my-asg
```

Resultado de ejemplo.

```
{
    "WarmPoolConfiguration": {
        "MinSize": 0,
        "PoolState": "Stopped"
    },
    "Instances": [
```

```
        {
            "InstanceId": "i-0b5e5e7521cfaa46c",
            "InstanceType": "t2.micro",
            "AvailabilityZone": "us-west-2a",
            "LifecycleState": "Warmed:Stopped",
            "HealthStatus": "Healthy",
            "LaunchTemplate": {
                "LaunchTemplateId": "lt-08c4cd42f320d5dcd",
                "LaunchTemplateName": "my-template-for-auto-scaling",
                "Version": "1"
            }
        },
        {
            "InstanceId": "i-0e21af9dcfb7aa6bf",
            "InstanceType": "t2.micro",
            "AvailabilityZone": "us-west-2a",
            "LifecycleState": "Warmed:Stopped",
            "HealthStatus": "Healthy",
            "LaunchTemplate": {
                "LaunchTemplateId": "lt-08c4cd42f320d5dcd",
                "LaunchTemplateName": "my-template-for-auto-scaling",
                "Version": "1"
            }
        }
    ]
}
```

Para ver el motivo de los errores en las comprobaciones de estado (AWS CLI)

Utilice el siguiente comando describe-scaling-activities.

```
aws autoscaling describe-scaling-activities --auto-scaling-group-name my-asg
```

A continuación se muestra un ejemplo de respuesta, donde `Description` indica que el grupo de escalado automático ha terminado una instancia, y `Cause` indica el motivo del error en la comprobación de estado.

Las actividades de escalado se ordenan por hora de inicio. En primer lugar, se describen las actividades aún en curso.

```
{
  "Activities": [
```

```
    {
      "ActivityId": "4c65e23d-a35a-4e7d-b6e4-2eaa8753dc12",
      "AutoScalingGroupName": "my-asg",
      "Description": "Terminating EC2 instance: i-04925c838b6438f14",
      "Cause": "At 2021-04-01T21:48:35Z an instance was taken out of service in
 response to EBS volume health check failure.",
      "StartTime": "2021-04-01T21:48:35.859Z",
      "EndTime": "2021-04-01T21:49:18Z",
      "StatusCode": "Successful",
      "Progress": 100,
      "Details": "{\"Subnet ID\":\"subnet-5ea0c127\",\"Availability Zone\":\"us-west-2a
\"...}",
      "AutoScalingGroupARN": "arn:aws:autoscaling:us-
west-2:123456789012:autoScalingGroup:283179a2-
f3ce-423d-93f6-66bb518232f7:autoScalingGroupName/my-asg"
    },
...
  ]
}
```

Ejemplos de cómo crear y gestionar piscinas termales con el AWS CLI

Puede crear y administrar piscinas climatizadas mediante los SDK Consola de administración de AWS, AWS Command Line Interface (AWS CLI) o los SDK.

En los siguientes ejemplos, se muestra cómo crear y administrar grupos de calentamiento con la AWS CLI.

Contenido

- Ejemplo 1: Mantener las instancias en estado detenido
- Ejemplo 2: Mantener las instancias en estado de ejecución
- Ejemplo 3: Mantener las instancias en estado de hibernación
- Ejemplo 4: devolver instancias al grupo de calentamiento al reducir horizontalmente
- Ejemplo 5: especificar el número mínimo de instancias en el grupo de calentamiento
- Ejemplo 6: definir el tamaño del grupo de calentamiento mediante una especificación personalizada
- Ejemplo 7: definir un tamaño absoluto de grupo de calentamiento
- Ejemplo 8: eliminar un grupo de calentamiento

Ejemplo 1: Mantener las instancias en estado **detenido**

El siguiente ejemplo put-warm-pool crea un grupo de calentamiento que mantiene las instancias en un estado `Stopped`.

```
aws autoscaling put-warm-pool --auto-scaling-group-name my-asg /
  --pool-state Stopped
```

Ejemplo 2: Mantener las instancias en **estado de** ejecución

El siguiente ejemplo put-warm-pool crea un grupo de calentamiento que mantiene las instancias en un estado `Running` en lugar de un estado `Stopped`.

```
aws autoscaling put-warm-pool --auto-scaling-group-name my-asg /
  --pool-state Running
```

Ejemplo 3: Mantener las instancias en **estado de hibernación**

El siguiente ejemplo put-warm-pool crea un grupo de calentamiento que mantiene las instancias en un estado `Hibernated` en lugar de un estado `Stopped`. Esto le permite detener instancias sin eliminar el contenido de su memoria (RAM).

```
aws autoscaling put-warm-pool --auto-scaling-group-name my-asg /
  --pool-state Hibernated
```

Ejemplo 4: devolver instancias al grupo de calentamiento al reducir horizontalmente

El siguiente ejemplo put-warm-pool crea un grupo de calentamiento que mantiene las instancias en un estado `Stopped` e incluye la opción `--instance-reuse-policy`. El valor de la política de reutilización de instancias `'{"ReuseOnScaleIn": true}'` le indica a Amazon EC2 Auto Scaling que devuelva las instancias al grupo de calentamiento cuando el grupo de escalado automático se reduce horizontalmente.

```
aws autoscaling put-warm-pool --auto-scaling-group-name my-asg /
  --pool-state Stopped --instance-reuse-policy '{"ReuseOnScaleIn": true}'
```

Ejemplo 5: especificar el número mínimo de instancias en el grupo de calentamiento

El siguiente ejemplo put-warm-pool crea un grupo de calentamiento que mantiene un mínimo de 4 instancias, de modo que haya al menos 4 instancias disponibles para gestionar los picos de tráfico.

```
aws autoscaling put-warm-pool --auto-scaling-group-name my-asg /
  --pool-state Stopped --min-size 4
```

Ejemplo 6: definir el tamaño del grupo de calentamiento mediante una especificación personalizada

De forma predeterminada, Amazon EC2 Auto Scaling administra el tamaño del grupo de calentamiento como la diferencia entre la capacidad máxima del grupo de escalado automático y la capacidad deseada. Sin embargo, puede administrar el tamaño del grupo de calentamiento independientemente de la capacidad máxima del grupo mediante la opción `--max-group-prepared-capacity`.

El siguiente ejemplo de put-warm-pool crea un grupo de calentamiento y establece la cantidad máxima de instancias que pueden existir simultáneamente tanto en el grupo de calentamiento como en el grupo de escalado automático. Si el grupo tiene una capacidad deseada de 800, el grupo de calentamiento tendrá inicialmente un tamaño de 100, ya que se inicializará tras ejecutar este comando.

```
aws autoscaling put-warm-pool --auto-scaling-group-name my-asg /
  --pool-state Stopped --max-group-prepared-capacity 900
```

Para mantener un número mínimo de instancias en el grupo de calentamiento, incluya la opción `--min-size` con el comando, de la siguiente manera.

```
aws autoscaling put-warm-pool --auto-scaling-group-name my-asg /
  --pool-state Stopped --max-group-prepared-capacity 900 --min-size 25
```

Ejemplo 7: definir un tamaño absoluto de grupo de calentamiento

Si configura el mismo valor para las opciones `--max-group-prepared-capacity` y `--min-size`, el grupo de calentamiento tiene un tamaño absoluto. El siguiente ejemplo put-warm-pool crea un grupo de calentamiento que mantiene un tamaño constante de grupo de calentamiento de 10 instancias.

```
aws autoscaling put-warm-pool --auto-scaling-group-name my-asg /
  --pool-state Stopped --min-size 10 --max-group-prepared-capacity 10
```

Ejemplo 8: eliminar un grupo de calentamiento

Utilice el siguiente comando delete-warm-pool para eliminar un grupo de calentamiento.

```
aws autoscaling delete-warm-pool --auto-scaling-group-name my-asg
```

Si hay instancias en el grupo de calentamiento, o si hay en curso actividades de escalado, utilice el comando delete-warm-pool con la opción `--force-delete`. Esta opción también termina las instancias de Amazon EC2 y cualquier acción pendiente del ciclo de vida.

```
aws autoscaling delete-warm-pool --auto-scaling-group-name my-asg --force-delete
```

Cambio zonal del grupo Auto Scaling

El cambio zonal es una capacidad del Amazon Application Recovery Controller (ARC). Con el cambio zonal, puede recuperarse rápidamente de las deficiencias de las aplicaciones en una zona de disponibilidad con una sola acción. Cuando habilita el cambio zonal para un grupo de Auto Scaling, el grupo se registra en el servicio de cambio zonal ARC. A continuación, puede iniciar un cambio zonal mediante la Consola de administración de AWS AWS CLI, o API y el grupo Auto Scaling considerará que la zona de disponibilidad con un cambio zonal activo está alterada.

Conceptos de cambio zonal del grupo Auto Scaling

Antes de continuar, asegúrese de estar familiarizado con los siguientes conceptos básicos relacionados con la integración con el cambio zonal de ARC.

Cambio zonal ARC

> Auto Scaling puede registrar grupos de Auto Scaling con cambio zonal ARC cuando se habilita esta función. Tras registrarse, podrá ver sus recursos con la `ListManagedResources` API ARC. Para obtener más información, consulte Cambio zonal en ARC en la Guía para desarrolladores de Amazon Application Recovery Controller (ARC).

Reequilibrio de la zona de disponibilidad

> Auto Scaling intenta mantener equilibrada la capacidad en cada zona de disponibilidad. Cuando se produce un desequilibrio entre las zonas de disponibilidad, Auto Scaling intenta corregirlo automáticamente. Para obtener más información, consulte Distribución de instancias.

Escalado dinámico

El escalado dinámico escala la capacidad deseada de su grupo de Auto Scaling en función de las métricas que elija con las políticas de escalado. Para obtener más información, consulte Escalado dinámico para Amazon EC2 Auto Scaling.

Comprobaciones de estado

Auto Scaling comprueba periódicamente el estado de todas las instancias de un grupo de Auto Scaling para asegurarse de que se estén ejecutando y en buenas condiciones. Cuando se detecta una instancia en mal estado, Auto Scaling la marca para reemplazarla. Para obtener más información, consulte Comprobaciones de estado para instancias en un grupo de escalado automático.

Actualización de instancias

Puede utilizar una actualización de instancias para actualizar las instancias del grupo de escalado automático. Una vez iniciada la actualización de una instancia, Auto Scaling intenta reemplazar todas las instancias de su grupo de Auto Scaling. Para obtener más información, consulte Uso de una actualización de instancias para actualizar instancias en un grupo de escalado automático.

Preescalado

Puede tolerar la pérdida de una sola zona de disponibilidad porque tiene suficiente capacidad en las zonas de disponibilidad restantes para su aplicación.

Escalado ascendente

Al aumentar la capacidad deseada de un grupo de Auto Scaling, Auto Scaling intenta lanzar instancias adicionales para cumplir con la nueva capacidad deseada. De forma predeterminada, Auto Scaling lanza la instancia de forma equilibrada para mantener la misma capacidad en cada zona de disponibilidad habilitada de un grupo de Auto Scaling.

Cómo funciona el cambio zonal para los grupos de Auto Scaling

Suponga que tiene un grupo de Auto Scaling con las siguientes zonas de disponibilidad:

- `us-east-1a`
- `us-east-1b`
- `us-east-1c`

Tiene activado el cambio zonal en todas las zonas de disponibilidad y observa errores en ellas, `us-east-1a` por lo que activa un cambio zonal. Los siguientes comportamientos se producen cuando se activa un cambio zonal. `us-east-1a`

- Ampliación: Auto Scaling lanzará todas las nuevas solicitudes de capacidad en las zonas de disponibilidad en buen estado (`us-east-1b`y`us-east-1c`).
- Escalado dinámico: Auto Scaling impedirá que las políticas de escalado reduzcan la capacidad deseada en todas las zonas de disponibilidad. Auto Scaling no impedirá que las políticas de escalado aumenten la capacidad deseada en todas las zonas de disponibilidad.
- Actualizaciones de instancias: Auto Scaling prolongará el tiempo de espera de cualquier proceso de actualización de instancias que se retrase mientras haya un cambio zonal activo.

En la siguiente tabla, se describe el comportamiento de las comprobaciones de estado de cada opción cuando se activa un cambio zonal. `us-east-1a`

Deterioro de la selección del comportamiento de las comprobaciones de estado de	Comportamiento de Health Check			
Reemplace lo poco saludable	Las instancias que parezcan estar en mal estado se reemplazarán en todas las zonas de disponibilidad (`us-east-1a us-east-1b` , y`us-east-1c`).			
Ignore el estado	Los casos que parezcan			

Deterioro de la selección del comportamiento de las comprobaciones de estado de	Comportamiento de Health Check			
	insalubres se sustituirán por `us-east-1b` y `us-east-1c` . Las instancias de la zona de disponibilidad no se reemplazarán con el cambio zonal activo (`us-east-1a`).			

Mejores prácticas para usar el cambio zonal

Para mantener una alta disponibilidad de sus aplicaciones al utilizar el cambio zonal, le recomendamos las siguientes prácticas recomendadas:

- Supervise EventBridge las notificaciones para determinar si se está produciendo un deterioro continuo de la zona de disponibilidad. Para obtener más información, consulte Se usa EventBridge para gestionar eventos de Auto Scaling.
- Utilice políticas de escalado con los umbrales adecuados para asegurarse de que tiene la capacidad suficiente para tolerar la pérdida de una zona de disponibilidad.
- Establezca una política de mantenimiento de instancias con un porcentaje mínimo de mantenimiento del 100%. Con esta configuración, Auto Scaling espera a que una nueva instancia esté lista para usarse antes de cerrar una instancia en mal estado.

Para los clientes preescalados, también recomendamos lo siguiente:

- Seleccione Ignorar el estado incorrecto como comportamiento de comprobación de estado para la zona de disponibilidad dañada, ya que no es necesario reemplazar la instancia en mal estado durante el caso de deterioro.
- Utilice el cambio automático zonal en ARC para sus grupos de Auto Scaling. La función de cambio automático zonal de ARC AWS permite desviar el tráfico de un recurso fuera de una zona de disponibilidad cuando se AWS detecta un deterioro en una zona de disponibilidad. Para obtener más información, consulte Cambio automático zonal en ARC en la Guía para desarrolladores de Amazon Application Recovery Controller (ARC).

Para los clientes con balanceadores de carga entre zonas desactivados, también recomendamos lo siguiente:

- Usa el balanceado solo para tu distribución por zonas de disponibilidad.
- Si está utilizando el cambio zonal tanto en los grupos de Auto Scaling como en los balanceadores de carga, cancele primero el cambio zonal en su grupo de Auto Scaling. A continuación, espere a que la capacidad se equilibre en todas las zonas de disponibilidad antes de cancelar el cambio zonal en el equilibrador de carga.
- Debido a la posibilidad de que la capacidad se desequilibre cuando se habilita el cambio zonal y se utiliza un balanceador de carga desactivado entre zonas, Auto Scaling incluye un paso de validación adicional. Si sigue las prácticas recomendadas, puede reconocer esta posibilidad seleccionando la Consola de administración de AWS casilla de verificación o utilizando la `skip-zonal-shift-validation` marca en, o. `CreateAutoScalingGroup` `UpdateAutoScalingGroup` `AttachTrafficSources`

Para obtener más información sobre el uso del cambio zonal con grupos de Auto Scaling, consulte el blog de AWS informática Cómo usar el cambio zonal con Amazon EC2 Auto Scaling.

Habilite el cambio zonal mediante el o el Consola de administración de AWS AWS CLI

Para activar el cambio zonal, utilice uno de los siguientes métodos.

Console

Para habilitar el cambio zonal en un grupo nuevo (consola)

1. Siga las instrucciones Creación de un grupo de Auto Scaling mediante una plantilla de lanzamiento y complete cada paso del procedimiento, hasta el paso 10.
2. En la página Integrar con otros servicios, para el cambio zonal del Application Recovery Controller (ARC), seleccione la casilla de verificación para habilitar el cambio zonal.
3. Para Comportamiento de Health Check, selecciona Ignorar lo que no es saludable o Reemplazar lo que no es saludable. Para obtener más información, consulte Cómo funciona el cambio zonal para los grupos de Auto Scaling.
4. Continúe con los pasos en Creación de un grupo de Auto Scaling mediante una plantilla de lanzamiento.

AWS CLI

Para habilitar el cambio zonal en un grupo nuevo ()AWS CLI

Agregue el parámetro `--availability-zone-impairment-policy` al comando create-auto-scaling-group.

El `--availability-zone-impairment-policy` parámetro tiene dos opciones:

- ZonalShiftEnabled— Si se establece en`true`, Auto Scaling registra el grupo de Auto Scaling con cambio zonal ARC y usted puede iniciar, actualizar o cancelar un cambio zonal en la consola ARC. Si se establece en`false`, Auto Scaling anula el registro del grupo Auto Scaling del cambio zonal de ARC. Debe tener ya activado el cambio zonal para configurarlo. `false`
- ImpairedZoneHealthCheckBehavior— Si se establece en esta `replace-unhealthy` opción, las instancias en mal estado se sustituirán en la zona de disponibilidad por el cambio zonal activo. Si se establece en esta `ignore-unhealthy` opción, las instancias en mal estado no se reemplazarán en la zona de disponibilidad con el cambio zonal activo. Para obtener más información, consulte Cómo funciona el cambio zonal para los grupos de Auto Scaling.

El siguiente ejemplo habilita el cambio zonal en un nuevo grupo de Auto Scaling denominado*my-asg*.

```
aws autoscaling create-auto-scaling-group \
  --launch-template LaunchTemplateName=my-launch-template,Version='1' \
```

```
--auto-scaling-group-name my-asg \
--min-size 1 \
--max-size 10 \
--desired-capacity 5 \
--availability-zones us-east-1a us-east-1b us-east-1c \
--availability-zone-impairment-policy '{
    "ZonalShiftEnabled": true,
    "ImpairedZoneHealthCheckBehavior": IgnoreUnhealthy
  }'
```

Console

Para habilitar el cambio zonal en un grupo existente (consola)

1. Abre la EC2 consola de Amazon en https://console.aws.amazon.com/ec2/y selecciona Auto Scaling Groups en el panel de navegación.
2. En la barra de navegación de la parte superior de la pantalla, elija la misma Región de AWS que utilizó cuando creó el grupo de escalado automático.
3. Seleccione la casilla situada junto al grupo de escalado automático.

   Se abre un panel dividido en la parte inferior de la página.
4. En la pestaña Integraciones, en el cambio zonal del Application Recovery Controller (ARC), seleccione Editar.
5. Seleccione la casilla de verificación para activar el cambio zonal.
6. Para Comportamiento de Health Check, selecciona Ignorar lo que no es saludable o Reemplazar lo que no es saludable. Para obtener más información, consulte Cómo funciona el cambio zonal para los grupos de Auto Scaling.
7. Elija Actualizar.

AWS CLI

Para habilitar el cambio zonal en un grupo existente ()AWS CLI

Agregue el parámetro `--availability-zone-impairment-policy` al comando update-auto-scaling-group.

El `--availability-zone-impairment-policy` parámetro tiene dos opciones:

- ZonalShiftEnabled— Si se establece en `true`, Auto Scaling registra el grupo de Auto Scaling con cambio zonal ARC y usted puede iniciar, actualizar o cancelar un cambio zonal en la consola ARC. Si se establece en `false`, Auto Scaling anula el registro del grupo Auto Scaling del cambio zonal de ARC. Debe tener ya activado el cambio zonal para configurarlo. `false`
- ImpairedZoneHealthCheckBehavior— Si se establece en esta `replace-unhealthy` opción, las instancias en mal estado se sustituirán en la zona de disponibilidad por el cambio zonal activo. Si se establece en esta `ignore-unhealthy` opción, las instancias en mal estado no se reemplazarán en la zona de disponibilidad con el cambio zonal activo. Para obtener más información, consulte Cómo funciona el cambio zonal para los grupos de Auto Scaling.

El siguiente ejemplo habilita el cambio zonal en el grupo de Auto Scaling especificado.

```
aws autoscaling update-auto-scaling-group --auto-scaling-group-name my-asg \
  --availability-zone-impairment-policy '{
      "ZonalShiftEnabled": true,
      "ImpairedZoneHealthCheckBehavior": IgnoreUnhealthy
    }'
```

Distribución de zonas de disponibilidad del grupo de escalado automático

La siguiente información describe las estrategias de zona de disponibilidad del grupo de escalado automático.

Estrategia de equilibrio de mejor esfuerzo

Auto Scaling mantiene una cantidad igual de instancias en todas las zonas de disponibilidad habilitadas. Si fallan los intentos de lanzamiento en una zona de disponibilidad, Auto Scaling intentará lanzar instancias en otra zona de disponibilidad en buen estado. Esta estrategia es importante para las aplicaciones que necesitan redundancia en las zonas de disponibilidad y no se ven afectadas por grupos desequilibrados.

Estrategia únicamente equilibrada

Auto Scaling mantiene una cantidad igual de instancias en todas las zonas de disponibilidad habilitadas. Si los intentos de lanzamiento fallan en una zona de disponibilidad, Auto Scaling seguirá intentando lanzar instancias en la zona de disponibilidad. Esta estrategia es importante para cumplir con ciertos requisitos, como las cargas de trabajo basadas en quórum o si su grupo

de escalado automático puede tolerar la pérdida de una zona de disponibilidad porque tiene suficiente capacidad en las zonas de disponibilidad restantes.

Entonces, las reservas se equilibran

Auto Scaling primero lanza las instancias en las reservas de capacidad asociadas al grupo, priorizando el uso de las reservas sobre la distribución uniforme en las zonas de disponibilidad habilitadas del grupo. Una vez completadas todas las reservas disponibles, Auto Scaling distribuye la capacidad restante de manera uniforme entre las zonas de disponibilidad habilitadas. Utilice esta estrategia para las cargas de trabajo sensibles a los costes que utilizan reservas de capacidad, como las cargas de trabajo de GPU, computación de alto rendimiento (HPC) o aprendizaje automático. Con esta estrategia, maximizar el uso de las reservas tiene prioridad incluso sobre la distribución por zonas de disponibilidad.

Para especificar las reservas que se deben priorizar, debe proporcionar un grupo de reservas de capacidad o un identificador de reserva de capacidad individual. Puede especificar el objetivo en su plantilla de lanzamiento o en el grupo Auto Scaling utilizando CapacityReservationTarget. Para priorizar más de una reserva, puede crear un grupo de recursos de reserva de capacidad. Esta estrategia admite las reservas On-Demand de capacidad, los bloques de capacidad y las reservas de capacidad interrumpibles.

> **Note**
>
> Durante el escalado, Auto Scaling elimina primero las instancias que no están en una reserva para preservar la utilización de la reserva. Si se requiere una mayor escalabilidad, Auto Scaling elimina las instancias reservadas, empezando por la zona de disponibilidad que tiene la mayoría de las instancias reservadas.

La selección de la estrategia de distribución de la zona de disponibilidad se encuentra en la sección Red del Consola de administración de AWS o puede utilizar los comandos create-auto-scaling-group o update-autoscaling-group.

Al elegir la estrategia de reservas y luego balanceada, también debe especificar las reservas de capacidad a las que desea priorizar, ya sea por grupo de reservas de capacidad o por ID de reserva de capacidad individual.

Para obtener más información, consulte Creación de grupos de escalado automático mediante plantillas de lanzamiento.

Desconexión o asociación de instancias en el grupo de escalado automático

Puede desconectar instancias del grupo de escalado automático. Después de separar una instancia, esa instancia se vuelve independiente y puede administrarse de manera independiente o asociarse a un grupo de escalado automático distinto al grupo original al que pertenecía. Esto puede resultar útil, por ejemplo, cuando desee realizar pruebas con instancias existentes que ya estén ejecutando su aplicación.

En este tema se proporcionan instrucciones sobre cómo desconectar y asociar instancias. Cuando asocia instancias, también puede usar una instancia existente en lugar de una desconectada.

En lugar de desconectar y reasociar una instancia al mismo grupo, le recomendamos que utilice el procedimiento de espera para eliminar temporalmente la instancia del grupo. Para obtener más información, consulte Eliminación temporal de las instancias de un grupo de escalado automático.

Contenido

- Consideraciones sobre la desconexión de instancias
- Consideraciones sobre la asociación de instancias
- Mueva una instancia a un grupo diferente mediante la opción de desconectar y asociar

Consideraciones sobre la desconexión de instancias

Cuando desconecta instancias, tenga en cuenta lo siguiente:

- Puede desasociar una instancia solo cuando está en estado `InService` o `StandBy`. Si va a desasociar instancias que están en el estado `StandBy`, tenga cuidado. Es posible que la inclusión del indicador `ShouldDecrementDesiredCapacity` en la llamada a la API cuando se intenta desasociar instancias después de colocarlas en el estado `StandBy` provoque que otras instancias terminen inesperadamente.
- Después de desconectar una instancia, esta sigue en funcionamiento y produce gastos. Para evitar gastos innecesarios, asegúrese de volver a asociar o finalizar las instancias desconectadas cuando ya no sean necesarias.
- Puede optar por reducir la capacidad deseada en el número de instancias que se van a desconectar. Si elige no reducir la capacidad, Amazon EC2 Auto Scaling lanza nuevas instancias para sustituir a las que ha desconectado con el propósito de mantener la capacidad deseada.

- Si la cantidad de instancias que está desconectando reduce el tamaño del grupo de escalado automático por debajo de su capacidad mínima, deberá reducir la capacidad mínima.
- Si desconecta varias instancias de la misma zona de disponibilidad sin reducir la capacidad deseada, el grupo se equilibrará a menos que suspenda el proceso `AZRebalance`. Para obtener más información, consulte Suspensión y reanudación de procesos de Amazon EC2 Auto Scaling.
- Si desconecta una instancia de un grupo de escalado automático que tiene un grupo de destino del balanceador de carga o un Classic Load Balancer, se cancela el registro de la instancia del balanceador de carga. Si está habilitado el drenaje de conexiones (retraso de anulación del registro) para el equilibrador de carga, Amazon EC2 Auto Scaling espera a que se completen las solicitudes en tránsito.

Consideraciones sobre la asociación de instancias

Tenga en cuenta lo siguiente al asociar instancias:

- Amazon EC2 Auto Scaling trata a las instancias asociadas de la misma manera que las instancias iniciadas por el grupo. Esto significa que las instancias asociadas se pueden finalizar durante los eventos de reducción horizontal si se seleccionan. Los permisos que otorga el rol AWSServiceRoleForAutoScaling vinculado a servicios permiten a Amazon EC2 Auto Scaling hacerlo.
- Cuando se asocian instancias, aumenta la capacidad deseada del grupo en el número de instancias que se asocian. Si la capacidad deseada más las instancias nuevas supera el tamaño máximo del grupo, la solicitud de asociación de instancias fallará.
- Si agrega instancias a su grupo, lo que provoca una distribución desigual entre las zonas de disponibilidad, Amazon EC2 Auto Scaling reequilibra el grupo para restablecer una distribución uniforme, a menos que suspenda el proceso de `AZRebalance`. Para obtener más información, consulte Suspensión y reanudación de procesos de Amazon EC2 Auto Scaling.
- Si asocia una instancia a un grupo de escalado automático que tiene un grupo de destino de balanceador de carga o un Classic Load Balancer, se registra la instancia con el balanceador de carga.

La instancia que desea asociar debe cumplir los siguientes criterios:

- La instancia se encuentra en el estado `running` con Amazon EC2.
- La AMI que se utiliza para lanzar la instancia debe existir.

- La instancia no es miembro de otro grupo de escalado automático.
- La instancia se inicia en una de las zonas de disponibilidad definidas en el grupo de escalado automático.
- Si el grupo de escalado automático tiene un grupo de destino de equilibrador de carga o equilibrador de carga clásico asociado, la instancia y el equilibrador de carga deben estar en la misma VPC.

Mueva una instancia a un grupo diferente mediante la opción de desconectar y asociar

Utilice uno de los siguientes procedimientos para desconectar una instancia del grupo de escalado automático y asociarla a otro grupo de escalado automático.

Para crear un nuevo grupo de escalado automático a partir de una instancia desconectada, consulte Cree un grupo de Auto Scaling a partir de una instancia existente mediante el AWS CLI (no se recomienda, crea una configuración de inicialización).

Console

Desasociación de una instancia de un grupo de escalado automático

1. Abra la consola Amazon EC2 en https://console.aws.amazon.com/ec2/y seleccione Auto Scaling Groups en el panel de navegación.
2. Seleccione la casilla situada junto al grupo de escalado automático.

   Se abre un panel dividido en la parte inferior de la página.
3. En la pestaña Administración de instancias, en Instancias, seleccione una instancia y elija Acciones, Desconectar.
4. En el cuadro de diálogo Desconectar la instancia, mantenga la casilla Reemplazar la instancia seleccionada para lanzar una instancia de reemplazo. Desactive la casilla de verificación para reducir la capacidad deseada.
5. Cuando se le pida la confirmación, escriba **`detach`** para confirmar la instancia especificada del grupo de escalado automático y, a continuación, elija Desconectar la instancia.

Ahora puede asociar la instancia a un grupo de escalado automático diferente.

Asociación de una instancia a un grupo de escalado automático

1. Abra la consola de Amazon EC2 en https://console.aws.amazon.com/ec2/.
2. (Opcional) En el panel de navegación, en Auto Scaling (Escalado automático), elija Auto Scaling Groups (Grupos de escalado automático). Seleccione el grupo de escalado automático y verifique que el tamaño máximo del grupo de escalado automático es lo suficientemente grande para poder agregar otra instancia. De lo contrario, en la pestaña Details (Detalles), aumente la capacidad máxima.
3. En el panel de navegación, en Instances (Instancias), elija Instances y seleccione una instancia.
4. Elija Acciones, Configuración de la instancia, Asociar a grupo de escalado automático.
5. En la página Attach to Auto Scaling Group (Asociar a grupo de escalado automático), en Auto Scaling Group (Grupo de Auto Scaling), seleccione el nombre del grupo de escalado automático y, a continuación, elija Attach (Asociar).
6. Si la instancia no cumple los criterios, recibirá un mensaje de error con los detalles. Por ejemplo, es posible que la instancia no esté en la misma zona de disponibilidad que el grupo de escalado automático. Seleccione Cerrar y vuelva a intentarlo con un grupo de escalado automático que cumpla los criterios.

AWS CLI

Para desconectar y asociar una instancia, utilice los siguientes comandos de ejemplo. Reemplace cada *user input placeholder* por su propia información.

Cómo desconectar una instancia de un grupo de escalado automático

1. Para describir las instancias actuales, utilice el siguiente comando describe-auto-scaling-instances.

```
aws autoscaling describe-auto-scaling-instances \
  --query 'AutoScalingInstances[?AutoScalingGroupName==`my-asg`]'
```

El siguiente ejemplo muestra los resultados del comando.

Anote el ID de la instancia que se va a eliminar del grupo. Este ID es necesario en el siguiente paso.

```
{
    "AutoScalingInstances": [
        {
            "ProtectedFromScaleIn": false,
            "AvailabilityZone": "us-west-2a",
            "LaunchTemplate": {
                "LaunchTemplateName": "my-launch-template",
                "Version": "1",
                "LaunchTemplateId": "lt-050555ad16a3f9c7f"
            },
            "InstanceId": "i-05b4f7d5be44822a6",
            "InstanceType": "t3.micro",
            "AutoScalingGroupName": "my-asg",
            "HealthStatus": "HEALTHY",
            "LifecycleState": "InService"
        },
        {
            "ProtectedFromScaleIn": false,
            "AvailabilityZone": "us-west-2a",
            "LaunchTemplate": {
                "LaunchTemplateName": "my-launch-template",
                "Version": "1",
                "LaunchTemplateId": "lt-050555ad16a3f9c7f"
            },
            "InstanceId": "i-0c20ac468fa3049e8",
            "InstanceType": "t3.micro",
            "AutoScalingGroupName": "my-asg",
            "HealthStatus": "HEALTHY",
            "LifecycleState": "InService"
        },
        {
            "ProtectedFromScaleIn": false,
            "AvailabilityZone": "us-west-2a",
            "LaunchTemplate": {
                "LaunchTemplateName": "my-launch-template",
                "Version": "1",
                "LaunchTemplateId": "lt-050555ad16a3f9c7f"
            },
            "InstanceId": "i-0787762faf1c28619",
            "InstanceType": "t3.micro",
            "AutoScalingGroupName": "my-asg",
            "HealthStatus": "HEALTHY",
            "LifecycleState": "InService"
```

```
        },
        {
            "ProtectedFromScaleIn": false,
            "AvailabilityZone": "us-west-2a",
            "LaunchTemplate": {
                "LaunchTemplateName": "my-launch-template",
                "Version": "1",
                "LaunchTemplateId": "lt-050555ad16a3f9c7f"
            },
            "InstanceId": "i-0f280a4c58d319a8a",
            "InstanceType": "t3.micro",
            "AutoScalingGroupName": "my-asg",
            "HealthStatus": "HEALTHY",
            "LifecycleState": "InService"
        }
    ]
}
```

2. Para desconectar una instancia sin reducir la capacidad deseada con el siguiente comando detach-instances.

```
aws autoscaling detach-instances --instance-ids i-05b4f7d5be44822a6 \
  --auto-scaling-group-name my-asg
```

Para desconectar una instancia y reducir la capacidad deseada, incluya la opción `--should-decrement-desired-capacity`.

```
aws autoscaling detach-instances --instance-ids i-05b4f7d5be44822a6 \
  --auto-scaling-group-name my-asg --should-decrement-desired-capacity
```

Ahora puede asociar la instancia a un grupo de escalado automático diferente.

Asociación de una instancia a un grupo de escalado automático

1. Para asociar una instancia a un grupo de escalado automático, utilice el siguiente comando attach-instances.

```
aws autoscaling attach-instances --instance-ids i-05b4f7d5be44822a6 --auto-
scaling-group-name my-asg-for-testing
```

2. Para verificar el tamaño del grupo de escalado automático después de asociar una instancia, utilice el siguiente comando describe-auto-scaling-groups.

```
aws autoscaling describe-auto-scaling-groups --auto-scaling-group-names my-asg-for-testing
```

El siguiente ejemplo de respuesta muestra que el grupo tiene dos instancias en ejecución, una de las cuales es la instancia que se ha asociado.

```
{
    "AutoScalingGroups": [
        {
            "AutoScalingGroupName": "my-asg-for-testing",
            "AutoScalingGroupARN": "arn",
            "LaunchTemplate": {
                "LaunchTemplateName": "my-launch-template",
                "Version": "2",
                "LaunchTemplateId": "lt-050555ad16a3f9c7f"
            },
            "MinSize": 1,
            "MaxSize": 5,
            "DesiredCapacity": 2,
            ...
            "Instances": [
                {
                    "ProtectedFromScaleIn": false,
                    "AvailabilityZone": "us-west-2a",
                    "LaunchTemplate": {
                        "LaunchTemplateName": "my-launch-template",
                        "Version": "1",
                        "LaunchTemplateId": "lt-050555ad16a3f9c7f"
                    },
                    "InstanceId": "i-05b4f7d5be44822a6",
                    "InstanceType": "t3.micro",
                    "HealthStatus": "Healthy",
                    "LifecycleState": "InService"
                },
                {
                    "ProtectedFromScaleIn": false,
                    "AvailabilityZone": "us-west-2a",
                    "LaunchTemplate": {
                        "LaunchTemplateName": "my-launch-template",
```

```
                        "Version": "2",
                        "LaunchTemplateId": "lt-050555ad16a3f9c7f"
                    },
                    "InstanceId": "i-00dcdfffdf5175890",
                    "InstanceType": "t3.micro",
                    "HealthStatus": "Healthy",
                    "LifecycleState": "InService"
                }
            ],
            ...
        }
    ]
}
```

Eliminación temporal de las instancias de un grupo de escalado automático

Puede poner una instancia que tiene el estado `InService` en el estado `Standby`, actualizar o resolver los problemas de la instancia y, a continuación, poner de nuevo la instancia en servicio. Las instancias que están a la espera siguen formando parte del grupo de escalado automático, pero no gestionan activamente el tráfico del balanceador de carga.

Esta característica le ayuda a detener y lanzar las instancias o a reiniciarlas sin preocuparse de que Amazon EC2 Auto Scaling termine las instancias como parte de sus comprobaciones de estado o durante eventos de reducción horizontal.

Por ejemplo, puede cambiar la instancia de Amazon Machine Image (AMI) de un grupo de escalado automático en cualquier momento cambiando la plantilla de lanzamiento o la configuración de lanzamiento. Cualquier instancia posterior que lance el grupo de escalado automático utilizará esta AMI. Sin embargo, el grupo de escalado automático no actualiza las instancias que están actualmente en servicio. Puede terminar estas instancias y permitir que Amazon EC2 Auto Scaling las reemplace, o bien utilizar la característica de actualización de instancias para terminar y reemplazar las instancias. O bien, puede poner las instancias en espera, actualizar el software y, a continuación, volver a poner las instancias en servicio.

Desconectar instancias de un grupo de escalado automático es similar a poner instancias en espera. La desconexión de instancias puede ser útil si desea asociar las instancias a un grupo diferente o administrarlas como instancias de EC2 independientes y, posiblemente, finalizarlas. Para

obtener más información, consulte Desconexión o asociación de instancias en el grupo de escalado automático.

Contenido

- Cómo funciona el estado en espera
- Consideraciones
- Estado de una instancia cuando está en espera
- Eliminación temporal de una instancia al ponerla en espera

Cómo funciona el estado en espera

El estado de espera funciona tal y como se indica a continuación para ayudarle a eliminar temporalmente una instancia del grupo de escalado automático:

1. Coloca una instancia en estado de espera. La instancia permanece en este estado hasta que suspenda el estado de espera.
2. Si hay un grupo de destino de balanceador de carga o un Classic Load Balancer asociados al grupo de escalado automático, se cancela el registro de la instancia del balanceador de carga. Si se habilita Connection Draining para el balanceador de carga, Elastic Load Balancing espera 300 segundos de forma predeterminada antes de completar el proceso de anulación del registro, para ayudar a que se completen las solicitudes en tránsito.
3. Puede actualizar la instancia o solucionar el problema.
4. Al salir del estado en espera, la instancia vuelve a estar en servicio.
5. Si hay un grupo de destino de balanceador de carga o un Classic Load Balancer asociados a un grupo de escalado automático, la instancia se registra en el balanceador de carga.

Para obtener más información sobre el ciclo de vida de las instancias de un grupo de escalado automático, consulte Ciclo de vida de instancias de Amazon EC2 Auto Scaling.

Consideraciones

A la hora de mover instancias al estado de espera y al sacarlas del estado de espera, se tienen en cuenta las siguientes consideraciones:

- Cuando pone una instancia en espera, puede reducir la capacidad deseada mediante esta operación o mantenerla en el mismo valor.

  - Si elige no reducir la capacidad deseada del grupo de escalado automático, Amazon EC2 Auto Scaling lanza una instancia para sustituir a la instancia que está en espera. La intención es ayudarlo a mantener la capacidad de su aplicación mientras una o más instancias están en espera.
  - Si elige reducir la capacidad deseada del grupo de escalado automático, se impide el lanzamiento de una instancia que sustituya a la instancia que está en espera.
- Después de volver a poner la instancia en servicio, la capacidad deseada se incrementa para reflejar el número de instancias que hay en el grupo de escalado automático.
- Para realizar el incremento (y la disminución), la nueva capacidad deseada debe estar entre el tamaño mínimo y máximo del grupo. De lo contrario, la operación no se llevará a cabo correctamente.
- Si en algún momento, después de poner una instancia en espera o de devolverla al servicio al salir del estado de espera, se descubre que su grupo de escalado automático no está equilibrado entre las zonas de disponibilidad, Amazon EC2 Auto Scaling lo compensa reequilibrando las zonas de disponibilidad, a menos que suspenda el proceso de `AZRebalance`. Para obtener más información, consulte Suspensión y reanudación de procesos de Amazon EC2 Auto Scaling.
- Se le cobrarán las instancias que se encuentran en estado de espera.

Estado de una instancia cuando está en espera

Amazon EC2 Auto Scaling no realiza comprobaciones de estado en las instancias que se encuentran en espera. Mientras la instancia se encuentra en un estado de espera, su estado de mantenimiento refleja el estado que tenía antes de que se pusiera en modo de espera. Amazon EC2 Auto Scaling no realiza una comprobación de estado en la instancia hasta que se vuelve a poner en servicio.

Por ejemplo, si pone una instancia que está en buen estado en modo de espera y después termina la instancia, Amazon EC2 Auto Scaling sigue informando que la instancia está en buen estado. Si intenta volver a poner en servicio una instancia terminada que estaba en espera, Amazon EC2 Auto Scaling realiza una comprobación de estado de la instancia, determina que ha terminado y no está en buen estado, y lanza una instancia de reemplazo. Para obtener más información, consulte Comprobaciones de estado para instancias en un grupo de escalado automático.

Eliminación temporal de una instancia al ponerla en espera

Use uno de los siguientes procedimientos para dejar una instancia fuera de servicio temporalmente colocándola en estado de espera.

Console

Para eliminar temporalmente una instancia

1. Abra la consola Amazon EC2 en https://console.aws.amazon.com/ec2/y seleccione Auto Scaling Groups en el panel de navegación.
2. Seleccione la casilla situada junto al grupo de escalado automático.

   Se abre un panel dividido en la parte inferior de la página.
3. En la pestaña Instance management (Administración de instancias), en Instances (Instancias), seleccione una instancia.
4. Seleccione Acciones, Establecer en En espera.
5. En el cuadro de diálogo Establecer en En espera, mantenga seleccionada la casilla de verificación Reemplazar instancia para lanzar una instancia de reemplazo. Desactive la casilla de verificación para reducir la capacidad deseada.
6. Cuando se solicite la confirmación, escriba **`standby`** para confirmar que se ha colocado la instancia especificada en el estado `Standby` y, a continuación, seleccione Establecer en En espera.
7. Puede actualizar la instancia o solucionar su problema según sea necesario. Cuando haya terminado, continúe con el siguiente paso para poner de nuevo la instancia en servicio.
8. Seleccione la instancia, elija Acciones y establezca en. InService En el cuadro de InService diálogo Definir como, seleccione Definir como InService.

AWS CLI

Para eliminar temporalmente una instancia del grupo de escalado automático, utilice los siguientes comandos de ejemplo. Reemplace cada *`user input placeholder`* por su propia información.

Para eliminar temporalmente una instancia

1. Utilice el siguiente comando describe-auto-scaling-instances para identificar la instancia que se va a actualizar.

```
aws autoscaling describe-auto-scaling-instances \
  --query 'AutoScalingInstances[?AutoScalingGroupName==`my-asg`]'
```

El siguiente ejemplo muestra los resultados del comando.

Anote el ID de la instancia que se va a eliminar del grupo. Este ID es necesario en el siguiente paso.

```
{
    "AutoScalingInstances": [
        {
            "ProtectedFromScaleIn": false,
            "AvailabilityZone": "us-west-2a",
            "LaunchTemplate": {
                "LaunchTemplateName": "my-launch-template",
                "Version": "1",
                "LaunchTemplateId": "lt-050555ad16a3f9c7f"
            },
            "InstanceId": "i-05b4f7d5be44822a6",
            "InstanceId": "t3.micro",
            "AutoScalingGroupName": "my-asg",
            "HealthStatus": "HEALTHY",
            "LifecycleState": "InService"
        },
       ...
    ]
}
```

2. Mueva la instancia a un estado `Standby` mediante el siguiente comando enter-standby. La opción `--should-decrement-desired-capacity` reduce la capacidad deseada para que el grupo de escalado automático no lance una instancia de reemplazo.

```
aws autoscaling enter-standby --instance-ids i-05b4f7d5be44822a6 \
  --auto-scaling-group-name my-asg --should-decrement-desired-capacity
```

A continuación, se muestra un ejemplo de respuesta.

```
{
    "Activities": [
        {
            "ActivityId": "3b1839fe-24b0-40d9-80ae-bcd883c2be32",
            "AutoScalingGroupName": "my-asg",
            "Description": "Moving EC2 instance to Standby:
 i-05b4f7d5be44822a6",
```

```
            "Cause": "At 2023-12-15T21:31:26Z instance i-05b4f7d5be44822a6 was
 moved to standby
              in response to a user request, shrinking the capacity from 4 to
 3.",
            "StartTime": "2023-12-15T21:31:26.150Z",
            "StatusCode": "InProgress",
            "Progress": 50,
            "Details": "{\"Subnet ID\":\"subnet-c934b782\",\"Availability Zone
\":\"us-west-2a\"}"
        }
    ]
}
```

3. (Opcional) Verifique que la instancia está `Standby` utilizando el siguiente comando describe-auto-scaling-instances.

```
aws autoscaling describe-auto-scaling-instances --instance-
ids i-05b4f7d5be44822a6
```

A continuación, se muestra un ejemplo de respuesta. Observe que el estado de la instancia ahora es `Standby`.

```
{
    "AutoScalingInstances": [
        {
            "ProtectedFromScaleIn": false,
            "AvailabilityZone": "us-west-2a",
            "LaunchTemplate": {
                "LaunchTemplateName": "my-launch-template",
                "Version": "1",
                "LaunchTemplateId": "lt-050555ad16a3f9c7f"
            },
            "InstanceId": "i-05b4f7d5be44822a6",
            "InstanceType": "t3.micro",
            "AutoScalingGroupName": "my-asg",
            "HealthStatus": "HEALTHY",
            "LifecycleState": "Standby"
        },
        ...
    ]
}
```

4. Puede actualizar la instancia o solucionar su problema según sea necesario. Cuando haya terminado, continúe con el siguiente paso para poner de nuevo la instancia en servicio.
5. Vuelva a poner la instancia en servicio usando el siguiente comando exit-standby.

```
aws autoscaling exit-standby --instance-ids i-05b4f7d5be44822a6 --auto-scaling-
group-name my-asg
```

A continuación, se muestra un ejemplo de respuesta.

```
{
    "Activities": [
        {
            "ActivityId": "db12b166-cdcc-4c54-8aac-08c5935f8389",
            "AutoScalingGroupName": "my-asg",
            "Description": "Moving EC2 instance out of Standby:
 i-05b4f7d5be44822a6",
            "Cause": "At 2023-12-15T21:46:14Z instance i-05b4f7d5be44822a6 was
 moved out of standby in
               response to a user request, increasing the capacity from 3 to
 4.",
            "StartTime": "2023-12-15T21:46:14.678Z",
            "StatusCode": "PreInService",
            "Progress": 30,
            "Details": "{\"Subnet ID\":\"subnet-c934b782\",\"Availability Zone
\":\"us-west-2a\"}"
        }
    ]
}
```

6. (Opcional) Compruebe que la instancia vuelve a estar en servicio utilizando el siguiente comando `describe-auto-scaling-instances`.

```
aws autoscaling describe-auto-scaling-instances --instance-
ids i-05b4f7d5be44822a6
```

A continuación, se muestra un ejemplo de respuesta. Observe que el estado de la instancia es `InService`.

```
{
    "AutoScalingInstances": [
        {
```

```
            "ProtectedFromScaleIn": false,
            "AvailabilityZone": "us-west-2a",
            "LaunchTemplate": {
                "LaunchTemplateName": "my-launch-template",
                "Version": "1",
                "LaunchTemplateId": "lt-050555ad16a3f9c7f"
            },
            "InstanceId": "i-05b4f7d5be44822a6",
            "InstanceType": "t3.micro",
            "AutoScalingGroupName": "my-asg",
            "HealthStatus": "HEALTHY",
            "LifecycleState": "InService"
        },
        ...
    ]
}
```

Eliminación de la infraestructura de Auto Scaling

Para eliminar completamente su infraestructura de escalado, realice las siguientes tareas.

Tareas

- Eliminar el grupo de Auto Scaling
- (Opcional) Eliminar la configuración de lanzamiento
- (Opcional) Eliminar la plantilla de lanzamiento
- (Opcional) Eliminar el balanceador de carga y los grupos de destino
- (Opcional) Elimine CloudWatch las alarmas
- Configure la protección contra la eliminación para sus recursos de Amazon EC2 Auto Scaling

Eliminar el grupo de Auto Scaling

Cuando elimina un grupo de Auto Scaling, su valores máximo, mínimo y deseado se establecen en 0. Como resultado, se terminan las instancias. La eliminación de una instancia también elimina todos los logs o datos asociados y los volúmenes de la instancia. Si no desea terminar una o varias instancias, puede desasociarlas antes de eliminar el grupo de Auto Scaling. Si el grupo tiene políticas de escalado, al eliminar el grupo se eliminarán las políticas, las acciones de alarma subyacentes y cualquier alarma que ya no tenga una acción asociada.

Para eliminar el grupo de Auto Scaling (consola)

1. Abra la consola Amazon EC2 en https://console.aws.amazon.com/ec2/y seleccione Auto Scaling Groups en el panel de navegación.
2. Seleccione la casilla de verificación situada junto al grupo de escalado automático y elija Acciones, Eliminar.
3. Cuando se le pida la confirmación, escriba **`delete`** para confirmar la eliminación del grupo de escalado automático especificado y, a continuación, elija Delete (Eliminar).

   Un icono de carga en la columna Name (Nombre) indica que el grupo de Auto Scaling se está eliminando. En las columnas Deseadas, Mín. y Máx. se muestran instancias `0` para el grupo de escalado automático. Se tarda unos minutos en terminar la instancia y eliminar el grupo. Actualice la lista para ver el estado actual.

Para eliminar su grupo de Auto Scaling (AWS CLI)

Utilice el siguiente comando delete-auto-scaling-group para eliminar el grupo de Auto Scaling. Esta operación no funciona si el grupo tiene instancias de EC2; solo es para grupos que no tienen ninguna instancia.

```
aws autoscaling delete-auto-scaling-group --auto-scaling-group-name my-asg
```

Si el grupo tiene instancias o actividades de escalado en curso, utilice el comando delete-auto-scaling-group con la opción `--force-delete`. Esto también terminará las instancias EC2. Al eliminar un grupo de escalado automático de la consola de Amazon EC2 Auto Scaling, la consola utiliza esta operación para terminar las instancias de EC2 y eliminar el grupo al mismo tiempo.

```
aws autoscaling delete-auto-scaling-group --auto-scaling-group-name my-asg --force-
delete
```

(Opcional) Eliminar la configuración de lanzamiento

Puede omitir este paso si desea mantener la configuración de lanzamiento para usarla en el futuro.

Para eliminar la configuración de lanzamiento (consola)

1. Abra la consola de Amazon EC2 en https://console.aws.amazon.com/ec2/.

2. En el panel de navegación izquierdo, en Escalado automático, elija Grupos de escalado automático.
3. Elija Configuraciones de lanzamiento cerca de la parte superior de la página. Cuando se le pida confirmación, elija Ver configuraciones de lanzamiento para confirmar que desea ver la página Configuraciones de lanzamiento.
4. Seleccione la configuración de lanzamiento y elija Acciones, Eliminar configuración de lanzamiento.
5. Cuando se le pida confirmación, seleccione Eliminar.

Para eliminar la configuración de inicio (AWS CLI)

Use el siguiente comando delete-launch-configuration.

```
aws autoscaling delete-launch-configuration --launch-configuration-name my-launch-config
```

(Opcional) Eliminar la plantilla de lanzamiento

Puede eliminar su plantilla de lanzamiento o simplemente una versión de su plantilla de lanzamiento. Al eliminar una plantilla de lanzamiento, todas sus versiones se eliminan.

Puede omitir este paso si desea mantener la plantilla de lanzamiento para usarla en el futuro.

Para eliminar la plantilla de lanzamiento (consola)

1. Abra la consola de Amazon EC2 en https://console.aws.amazon.com/ec2/.
2. En el panel de navegación, en Instances, seleccione Launch Templates.
3. Seleccione su plantilla de lanzamiento y, a continuación, realice una de las siguientes operaciones:

   - Elija Actions, Delete template. Cuando se le pida la confirmación, escriba **`Delete`** para confirmar la eliminación de la plantilla de lanzamiento especificada y, a continuación, elija Delete (Eliminar).
   - Elija Actions (Acciones), Delete template version (Eliminar plantilla de lanzamiento). Seleccione la versión que desea eliminar y elija Delete (Eliminar).

Para eliminar la plantilla de lanzamiento (AWS CLI)

Utilice el siguiente comando delete-launch-template para eliminar la plantilla y todas sus versiones.

```
aws ec2 delete-launch-template --launch-template-id lt-068f72b72934aff71
```

De forma alternativa, puede utilizar el comando delete-launch-template-versions para eliminar una versión específica de una plantilla de lanzamiento.

```
aws ec2 delete-launch-template-versions --launch-template-id lt-068f72b72934aff71 --
versions 1
```

(Opcional) Eliminar el balanceador de carga y los grupos de destino

Omita este paso si el grupo de Auto Scaling no está asociado a un balanceador de carga de Elastic Load Balancing o si desea conservar el balanceador de carga para usarlo en el futuro.

Para eliminar el balanceador de carga (consola)

1. Abra la consola de Amazon EC2 en https://console.aws.amazon.com/ec2/.
2. En el panel de navegación, en Equilibrio de carga, elija Equilibradores de carga.
3. Seleccione el balanceador de carga y elija Actions (Acciones), Delete (Eliminar).
4. Cuando se le indique que confirme, seleccione Yes, Delete (Sí, borrar).

Para eliminar el grupo de destino (consola)

1. En el panel de navegación, en Load Balancing (Equilibración de carga), elija Target Groups (Grupos de destino).
2. Elija el grupo de destino y elija Actions (Acciones), Delete (Eliminar).
3. Cuando se le indique que confirme, seleccione Yes, Delete (Sí, borrar).

Para eliminar el balanceador de carga asociado al grupo Auto Scaling (AWS CLI)

Para los balanceadores de carga de aplicaciones y los balanceadores de carga de red, utilice los siguientes comandos delete-load-balancer y delete-target-group.

```
aws elbv2 delete-load-balancer --load-balancer-arn my-load-balancer-arn
aws elbv2 delete-target-group --target-group-arn my-target-group-arn
```

En los balanceadores de carga clásicos, utilice el siguiente comando delete-load-balancer.

```
aws elb delete-load-balancer --load-balancer-name my-load-balancer
```

(Opcional) Elimine CloudWatch las alarmas

Para eliminar las CloudWatch alarmas asociadas a su grupo de Auto Scaling, complete los siguientes pasos. Por ejemplo, es posible que tenga alarmas asociadas a políticas de escalado simple o escalado por pasos.

> **Note**
>
> Al eliminar un grupo de Auto Scaling, se eliminan automáticamente las CloudWatch alarmas que Amazon EC2 Auto Scaling administra para una política de escalado de seguimiento de objetivos.

Puede omitir este paso si su grupo de Auto Scaling no está asociado a ninguna CloudWatch alarma o si desea conservar las alarmas para usarlas en el futuro.

Para eliminar las CloudWatch alarmas (consola)

1. Abra la CloudWatch consola en https://console.aws.amazon.com/cloudwatch/.
2. En el panel de navegación, elija Alarms.
3. Elija las alarmas y elija Action (Acción), Delete (Eliminar).
4. Cuando se le pida confirmación, seleccione Eliminar.

Para eliminar las CloudWatch alarmas (AWS CLI)

Utilice el comando delete-alarms. Puede eliminar una o más alarmas a la vez. Por ejemplo, utilice el siguiente comando para eliminar las alarmas `Step-Scaling-AlarmHigh-AddCapacity` y `Step-Scaling-AlarmLow-RemoveCapacity`.

```
aws cloudwatch delete-alarms --alarm-name Step-Scaling-AlarmHigh-AddCapacity Step-Scaling-AlarmLow-RemoveCapacity
```

Configure la protección contra la eliminación para sus recursos de Amazon EC2 Auto Scaling

Proteja su infraestructura de Amazon EC2 Auto Scaling de una eliminación accidental mediante la configuración de varios niveles de protección. Auto Scaling ofrece varios enfoques para evitar la eliminación no deseada de recursos en sus grupos de Auto Scaling y en las instancias de Amazon EC2 que administra.

Contenido

- Configurar la protección de eliminación de grupos de Auto Scaling
- Controle los permisos de eliminación con las políticas de IAM

Configurar la protección de eliminación de grupos de Auto Scaling

La protección contra la eliminación es una configuración a nivel de recursos que evita que el grupo de Amazon EC2 Auto Scaling se elimine accidentalmente. Cuando está habilitada, la protección de eliminación impide que la operación de la DeleteAutoScalingGroupAPI se lleve a cabo correctamente, lo que requiere que primero actualice la configuración de protección de eliminación a un nivel menos restrictivo antes de poder eliminar el grupo de Auto Scaling.

Amazon EC2 Auto Scaling ofrece tres niveles de protección contra la eliminación:

Ninguno (predeterminado)

No hay ninguna protección de eliminación habilitada, lo que significa que su grupo de Auto Scaling se puede eliminar con o sin usar la `ForceDelete` opción. Cuando `ForceDelete` se utilice, todas las instancias de Amazon EC2 administradas por su grupo de Auto Scaling también se cancelarán forzosamente sin ejecutar enlaces de ciclo de vida de terminación.

Evite la eliminación forzada

Tu grupo de Auto Scaling no se puede eliminar cuando usas `ForceDelete` esta opción. Esta configuración permite eliminar grupos de Auto Scaling vacíos (grupos sin instancias). Esta opción se recomienda para las cargas de trabajo de producción en las que se desee evitar la terminación masiva de instancias y, al mismo tiempo, permitir la limpieza de grupos vacíos.

Impida la eliminación total

Su grupo de Auto Scaling no se puede eliminar independientemente de si se utiliza la `ForceDelete` opción. Esta opción proporciona la mejor protección contra la eliminación

accidental. Es necesario deshabilitar explícitamente la protección contra la eliminación antes de que se pueda eliminar el grupo de Auto Scaling. Esto se recomienda para los grupos de Auto Scaling de misión crítica que se deben eliminar rara vez o nunca.

Cómo funciona la protección contra la eliminación

Cuando intentas operar con la DeleteAutoScalingGroupAPI con la protección contra la eliminación habilitada:

1. Amazon EC2 Auto Scaling valida la configuración de protección contra eliminación antes de procesar la solicitud.
2. Si el nivel de protección de eliminación configurado bloquea el intento de eliminación, Amazon EC2 Auto Scaling devuelve `ValidationError` un.
3. Su grupo de Auto Scaling y sus instancias de Amazon EC2 permanecen sin cambios.
4. Debe actualizar la configuración de protección contra la eliminación a un nivel menos restrictivo antes de poder eliminar su grupo de Auto Scaling.

La protección contra la eliminación no impide otras operaciones como:

- Actualización de la configuración del grupo de Auto Scaling.
- Finalización de instancias individuales.
- Operaciones de escalado (manuales o automáticas).
- Suspender o reanudar los procesos.

Para obtener más información sobre cómo gestionar correctamente la terminación de instancias, consulte. Diseño de aplicaciones para gestionar sin problemas la terminación de instancias

Configure la protección contra la eliminación

Puede configurar la protección contra la eliminación al crear un grupo de Auto Scaling o actualizar la configuración de un grupo de Auto Scaling existente.

Console

Para crear un grupo de Auto Scaling con protección contra eliminación

1. Abra la consola Amazon EC2 en https://console.aws.amazon.com/ec2/y seleccione Auto Scaling Groups en el panel de navegación.
2. Elija Create Auto Scaling group (Crear grupo de escalado automático).
3. Complete los pasos de configuración de su grupo de Auto Scaling.
4. En la página Configurar el tamaño y el escalado del grupo, expanda Ajustes adicionales.
5. Para la protección contra la eliminación de grupos de Auto Scaling, elija el nivel de protección que desee:
   - Ninguno: sin protección contra la eliminación (predeterminado)
   - Impedir la eliminación forzada: bloquea las operaciones de eliminación forzada
   - Impedir todas las eliminaciones: bloquear todas las operaciones de eliminación
6. Complete los pasos restantes para crear su grupo de Auto Scaling.

Para actualizar la protección contra la eliminación en un grupo de Auto Scaling existente

1. Abra la consola Amazon EC2 en https://console.aws.amazon.com/ec2/y seleccione Auto Scaling Groups en el panel de navegación.
2. Seleccione la casilla situada junto al grupo de escalado automático.
3. Seleccione Acciones, Editar.
4. En Configuración adicional, actualice la configuración de protección contra la eliminación de grupos de Auto Scaling.
5. Elija Actualizar.

AWS CLI

Para crear un grupo de Auto Scaling con protección contra eliminación

Utilice el comando create-auto-scaling-group con el parámetro: `--deletion-protection`

```
aws autoscaling create-auto-scaling-group \
    --auto-scaling-group-name my-asg \
    --launch-template LaunchTemplateName=my-template,Version='$Latest' \
```

```
    --min-size 1 \
    --max-size 5 \
    --desired-capacity 2 \
    --vpc-zone-identifier "subnet-12345678,subnet-87654321" \
    --deletion-protection prevent-force-deletion
```

Los valores válidos son: | | `--deletion-protection` none `prevent-force-deletion` `prevent-all-deletion`

Para actualizar la protección contra la eliminación en un grupo de Auto Scaling existente

Utilice el comando update-auto-scaling-group:

```
aws autoscaling update-auto-scaling-group \
    --auto-scaling-group-name my-asg \
    --deletion-protection prevent-all-deletion
```

Para deshabilitar la protección contra la eliminación

Defina la protección contra la eliminación ennone:

```
aws autoscaling update-auto-scaling-group \
    --auto-scaling-group-name my-asg \
    --deletion-protection none
```

Para verificar el estado de la protección contra la eliminación

Utilice el comando describe-auto-scaling-groups:

```
aws autoscaling describe-auto-scaling-groups \
    --auto-scaling-group-names my-asg
```

Controle los permisos de eliminación con las políticas de IAM

Utilice políticas AWS Identity and Access Management (IAM) para controlar qué usuarios y roles pueden eliminar grupos de Auto Scaling. IAM-basedlos controles proporcionan una capa adicional de seguridad al restringir los permisos a nivel de identidad.

Las políticas de IAM son especialmente útiles cuando se quiere:

- Permita a diferentes usuarios diferentes niveles de acceso a las operaciones de Auto Scaling.
- Impida que usuarios específicos usen la `ForceDelete` opción incluso si pueden realizar otras operaciones de Auto Scaling.
- Restrinja los permisos de eliminación a grupos específicos de Auto Scaling.

La siguiente política permite eliminar un grupo de escalado automático solo si el grupo tiene la etiqueta **`environment=development`**.

JSON

```
{
   "Version":"2012-10-17",
   "Statement": [{
      "Effect": "Allow",
      "Action": "autoscaling:DeleteAutoScalingGroup",
      "Resource": "*",
      "Condition": {
          "StringEquals": { "aws:ResourceTag/environment": "development" }
      }
   }]
}
```

La siguiente política utiliza la clave de `autoscaling:ForceDelete` condición para controlar el acceso a la acción de la `DeleteAutoScalingGroup` API. Esto puede impedir que algunos usuarios utilicen la `ForceDelete` operación, que termina todas las instancias de Amazon EC2 dentro de un grupo de Auto Scaling.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [{
        "Effect": "Deny",
        "Action": "autoscaling:DeleteAutoScalingGroup",
        "Resource": "*",
        "Condition": {
            "Bool": {
                "autoscaling:ForceDelete": "true"
```

```
            }
        }
    }]
}
```

Como alternativa, si no utiliza claves de condición para controlar el acceso a los grupos de escalado automático, puede especificar los ARN de los recursos del elemento `Resource` para controlar el acceso.

La siguiente política otorga a los usuarios permisos para usar la acción de `DeleteAutoScalingGroup` API, pero solo para los grupos de Auto Scaling cuyo nombre comience por**`devteam-`**.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "autoscaling:DeleteAutoScalingGroup",
            "Resource": "arn:aws:autoscaling:us-
east-1:111122223333:autoScalingGroup:*:autoScalingGroupName/devteam-*"
        }
    ]
}
```

También puede especificar varios ARN incluyéndolos en una lista. Al incluir el UUID, se garantiza que el acceso se conceda al grupo de escalado automático especificado. El UUID de un grupo nuevo es diferente del UUID de un grupo eliminado con el mismo nombre.

```
"Resource": [
    "arn:aws:autoscaling:region:account-
id:autoScalingGroup:uuid:autoScalingGroupName/devteam-1",
    "arn:aws:autoscaling:region:account-
id:autoScalingGroup:uuid:autoScalingGroupName/devteam-2",
    "arn:aws:autoscaling:region:account-
id:autoScalingGroup:uuid:autoScalingGroupName/devteam-3"
]
```

Para ver ejemplos adicionales de políticas de IAM para Amazon EC2 Auto Scaling, incluidas las políticas que controlan los permisos de eliminación, consulte. Ejemplos de políticas basadas en identidades

Reemplazo de las instancias del grupo de escalado automático

Amazon EC2 Auto Scaling ofrece capacidades que le permiten reemplazar las instancias de Amazon EC2 de su grupo de escalado automático después de realizar actualizaciones, como añadir nuevas plantillas de lanzamiento con una nueva imagen de máquina de Amazon (AMI) o nuevos tipos de instancias. También le ayuda a optimizar las actualizaciones, ya que le ofrece la opción de incluirlas en la misma operación que reemplaza a las instancias.

En esta sección se incluye información que le ayudará a hacer lo siguiente:

- Comenzar una actualización de instancias para reemplazar instancias en el grupo de Auto Scaling.
- Declarar actualizaciones específicas que describen una configuración deseada y actualizar el grupo de Auto Scaling a la configuración deseada.
- Omitir el reemplazo de instancias ya actualizadas.
- Utilice puntos de control para actualizar instancias en fases y realizar verificaciones en las instancias en puntos específicos.
- Utilice el tiempo de incorporación para hacer una pausa al final de una actualización de la instancia con el fin de validar el estado de la misma.
- Recibir notificaciones por correo electrónico cuando se llega a un punto de control.
- Utilizar una reversión para restaurar el grupo de escalado automático a la configuración que utilizaba con anterioridad.
- Se revierte automáticamente si la actualización de la instancia falla por algún motivo o si alguna CloudWatch alarma de Amazon que especifiques pasa a ese `ALARM` estado.
- Limitar la duración de las instancias para garantizar versiones de software y configuraciones de instancias coherentes en todo el grupo de escalado automático.
- Sustituya los volúmenes raíz sin cerrar las instancias y, al mismo tiempo, conserve las interfaces de red, los volúmenes no raíz y las políticas de IAM.

Contenido

- Uso de una actualización de instancias para actualizar instancias en un grupo de escalado automático
- Reemplazo de instancias de Auto Scaling en función de la duración máxima de la instancia

Uso de una actualización de instancias para actualizar instancias en un grupo de escalado automático

Puede utilizar una actualización de instancias para actualizar las instancias del grupo de escalado automático. Esta función puede resultar útil cuando un cambio de configuración requiere que reemplace las instancias o sus volúmenes raíz, especialmente si su grupo de Auto Scaling contiene una gran cantidad de instancias.

Algunas situaciones en las que una actualización de instancias puede ayudar son las siguientes:

- Implementación de una nueva imagen de máquina de Amazon (AMI) o script de datos de usuario en todo el grupo de escalado automático. Puede crear una nueva plantilla de lanzamiento con los cambios y, a continuación, utilizar una actualización de instancias para implementar las actualizaciones de forma inmediata.
- Migración de instancias a nuevos tipos de instancias para aprovechar las últimas mejoras y optimizaciones.
- Cambio en los grupos de escalado automático para que pasen de utilizar una configuración de lanzamiento a una plantilla de lanzamiento. Puede copiar las configuraciones de lanzamiento en las plantillas de lanzamiento y, a continuación, utilizar una actualización de instancias para actualizar las instancias a las nuevas plantillas. Para obtener más información acerca de la migración a plantillas de lanzamiento, consulte Migración de sus grupos de escalado automático a plantillas de lanzamiento.
- Aplica parches de seguridad o actualizaciones de software y, al mismo tiempo, preserva el estado de ejecución prolongado de las instancias y evita las restricciones de capacidad con tipos de instancias especializados, como las instancias de GPU o Mac.

Contenido

- Funcionamiento de una actualización de instancias en un grupo de escalado automático
- Comprensión de los valores predeterminados de una actualización de instancias
- Inicie la actualización de una instancia con Consola de administración de AWS o AWS CLI
- Supervise la actualización de una instancia mediante Consola de administración de AWS o AWS CLI
- Reemplace los volúmenes raíz durante la actualización de la instancia
- Cancela la actualización de una instancia mediante Consola de administración de AWS o AWS CLI

- Inversión de cambios con una reversión manual o automática
- Uso de una actualización de instancias con la omisión de coincidencias
- Agregar puntos de control a una actualización de instancias

Funcionamiento de una actualización de instancias en un grupo de escalado automático

En este tema, se describe cómo funciona una actualización de instancias y se introducen los conceptos clave que necesita conocer para utilizarla de forma eficaz.

Contenido

- Funcionamiento
- Conceptos clave
- Periodo de gracia de la comprobación de estado
- Compatibilidad de los tipos de instancias
- Limitaciones

Funcionamiento

Para actualizar las instancias de un grupo de escalado automático, puede definir una nueva configuración que contenga la última versión de la aplicación y cualquier otra actualización que desee realizar.

La actualización de instancias admite dos estrategias para actualizar las instancias:

- Estrategia continua (predeterminada): finaliza las instancias y lanza otras nuevas en lotes según sus preferencias. Esto garantiza que su grupo de Auto Scaling mantenga la capacidad y disponibilidad deseadas durante todo el proceso de actualización.
- Estrategia de reemplazo del volumen raíz: actualiza las instancias reemplazando solo el volumen raíz sin terminar la instancia. Esto preserva las interfaces de red de las instancias, los volúmenes de EBS que no son raíz y los datos del almacén de instancias.

Requisitos para la estrategia de reemplazo del volumen raíz:

- Su grupo de Auto Scaling debe usar una política de instancias mixtas

- Todas las anulaciones de la política de instancias mixtas deben especificar una `ImageId`
- AMIs debe contener solo un volumen raíz
- Todas las instancias deben coincidir con la configuración de la plantilla de lanzamiento del grupo
- Debe iniciar la actualización de la instancia con la configuración deseada que tenga una política de instancias mixtas que contenga `ImageId` anulaciones.

Cuando la opción Omitir coincidencia está habilitada, Auto Scaling compara la ID de AMI actual de cada instancia con IDs la AMI de la configuración deseada. Solo reemplaza las instancias en las que la AMI IDs no coincide, lo que le permite omitir las instancias que ya están actualizadas.

Realice una actualización de instancias

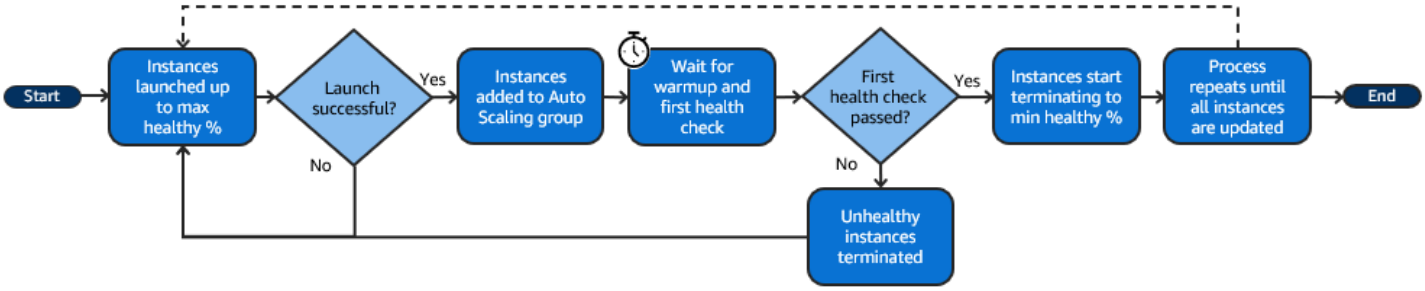
Inicia una actualización de instancias para reemplazar las instancias existentes por otras nuevas en función de esa configuración.

1. Cree una nueva plantilla de lanzamiento o actualice la existente con los cambios de configuración deseados, como una nueva imagen de máquina de Amazon (AMI). Para obtener más información, consulte Creación de una plantilla de lanzamiento para un grupo de Auto Scaling.
2. Inicie la actualización de la instancia mediante la consola AWS CLI Amazon EC2 Auto Scaling o el SDK:
   - Especifique la nueva plantilla de lanzamiento o versión de la plantilla de lanzamiento que ha creado. Se utilizará para lanzar nuevas instancias.
   - Establezca los porcentajes de buen estado mínimo y máximo. Esto controla cuántas instancias se reemplazan simultáneamente y si se lanzan nuevas instancias antes de terminar las antiguas.
   - Configure los ajustes opcionales, como los siguientes:
     - Puntos de control: pausa la actualización de la instancia después de un porcentaje determinado de reemplazos para comprobar el progreso.
     - Tiempo de incorporación: haga una pausa al final de la actualización de una instancia para validar el estado de la instancia antes de que se dé por finalizada su actualización.
     - Omitir la coincidencia: compara las instancias antiguas con la nueva configuración y reemplaza solo las que no coincidan. Al iniciar una actualización de instancias desde la consola, la omisión de coincidencias está activada de forma predeterminada.
     - Tipos de instancias múltiples: aplique una política de instancias mixtas nueva o actualizada como parte de la configuración deseada.

Cuando se haya iniciado la actualización de instancias, Amazon EC2 Auto Scaling hará lo siguiente:

- Reemplazar las instancias en lotes en función de los porcentajes de buen estado mínimo y máximo.
- Lanzar primero las instancias nuevas antes de terminar las antiguas si el porcentaje de buen estado mínimo está establecido en 100 por ciento. Esto garantiza que la capacidad deseada se mantenga en todo momento.
- Compruebe el estado de las instancias y espere a que se calienten antes de reemplazar más instancias.
- Termine y reemplace las instancias que no estén en buen estado.
- Actualice automáticamente la configuración del grupo de escalado automático con los nuevos cambios de configuración una vez que la actualización de la instancia se haya realizado correctamente.
- Sustituya las instancias `InService` antes de las que estén en un pool caliente.

El siguiente diagrama de flujo ilustra el comportamiento que ocurre si se lanzan instancias antes de terminarlas cuando se establece el porcentaje de buen estado mínimo en 100 por ciento.



> **Note**
>
> Es necesario especificar los porcentajes de estado máximos para una actualización de instancias solo si no ha establecido una política de mantenimiento de instancias o si necesita anular una ya existente. Para obtener más información, consulte Políticas de mantenimiento de instancias.
>
> Del mismo modo, solo necesita especificar el período de preparación de instancias para una actualización de instancias solo si no ha habilitado a preparación predeterminada o si necesita anular la configuración predeterminada. Para obtener más información, consulte Establecimiento de la preparación predeterminada de instancias para un grupo de escalado automático.

Conceptos clave

Antes de empezar, familiarícese con los siguientes conceptos principales de actualización de instancias:

Porcentaje de buen estado mínimo

El porcentaje de buen estado mínimo es el porcentaje de la capacidad que se desea mantener en servicio, en buen estado y lista para usar durante una actualización de instancias, de modo que esta pueda continuar. Por ejemplo, si el porcentaje de buen estado mínimo es del 90 por ciento y el porcentaje máximo en buen estado es del 100 por ciento, se reemplazará el 10 por ciento de capacidad por vez. Si las nuevas instancias no superan las comprobaciones de estado, Amazon EC2 Auto Scaling las finaliza y reemplaza. Si la actualización de instancias no puede lanzar ninguna instancia en buen estado, la actualización de instancias acabará fallando y el 90 por ciento restante del grupo permanecerá intacto. Si las nuevas instancias se mantienen en buen estado y completan el periodo de preparación, Amazon EC2 Auto Scaling podrá seguir reemplazando otras instancias.

Una actualización de instancias puede reemplazar las instancias de una en una, varias a la vez o todas a la vez. Para reemplazar una instancia de una en una, establezca el porcentaje mínimo y máximo en buen estado en el 100 por ciento. Esto cambia el comportamiento de la actualización de instancias para lanzarla antes de su finalización, lo que evita que la capacidad del grupo caiga por debajo del 100 por ciento de la capacidad deseada. Para reemplazar todas las instancias a la vez, establezca el porcentaje de buen estado mínimo en el 0 por ciento.

Porcentaje máximo en buen estado

El porcentaje máximo en buen estado de estado es el porcentaje de la capacidad deseada al que su grupo de escalado automático puede aumentar al reemplazar instancias. La diferencia entre el mínimo y el máximo no puede ser superior a 100. Un rango mayor aumenta la cantidad de instancias que se pueden reemplazar al mismo tiempo.

Preparación de las instancias

La preparación de las instancias es el periodo de tiempo desde que el estado de una nueva instancia cambia a `InService` hasta que se considera que ha finalizado la inicialización. Durante la actualización de instancias, en caso de que estas superen las comprobaciones de estado, Amazon EC2 Auto Scaling no procede inmediatamente a reemplazar la siguiente instancia tras determinar que una instancia recién lanzada está en buen estado. Espera durante el periodo de preparación antes de pasar a reemplazar otras instancias. Esto puede resultar útil cuando

la aplicación aún necesita algo de tiempo de inicialización antes de que pueda responder a las solicitudes.

La preparación de instancias funciona de la misma manera que la preparación de instancias predeterminada. Por lo tanto, se aplican las mismas consideraciones de escalado. Para obtener más información, consulte Establecimiento de la preparación predeterminada de instancias para un grupo de escalado automático.

Configuración deseada

La configuración deseada es la nueva configuración que quiere que Amazon EC2 Auto Scaling implemente en el grupo de escalado automático. Por ejemplo, puede especificar una nueva plantilla de lanzamiento y nuevos tipos de instancias. Durante una actualización de instancias, Amazon EC2 Auto Scaling actualiza el grupo de Auto Scaling a la configuración deseada. Si se produce un evento de escalado horizontal durante la actualización de instancias, Amazon EC2 Auto Scaling lanza nuevas instancias con la configuración deseada en lugar de la configuración actual del grupo. Una vez que la actualización de instancias se realice correctamente, Amazon EC2 Auto Scaling actualiza la configuración del grupo de escalado automático para reflejar la nueva configuración deseada que ha especificado como parte de la actualización de instancias.

Omisión de coincidencias

La opción de omisión de coincidencias le dice a Amazon EC2 Auto Scaling que ignore las instancias que ya tienen sus actualizaciones más recientes. De esta forma, no reemplaza más instancias de las necesarias. Esto es útil cuando quiere asegurarse de que su grupo de escalado automático utilice una versión determinada de la plantilla de lanzamiento y solo sustituya las instancias que usan una versión diferente.

Puntos de control

Un punto de comprobación es un punto en el tiempo en el que la actualización de instancias se detiene durante un periodo de tiempo especificado. Una actualización de instancias puede contener varios puntos de control. Amazon EC2 Auto Scaling emite eventos para cada punto de control. Por lo tanto, puede añadir una EventBridge regla para enviar los eventos a un destino, como Amazon SNS, para que se le notifique cuando se alcance un punto de control. Una vez llegado a un punto de control, podrá verificar la implementación. Si se identifica algún problema, puede cancelar la actualización de instancias o revertirla. La capacidad de implementar actualizaciones en fases es un beneficio clave de los puntos de control. Si no utiliza puntos de control, los reemplazos sucesivos se realizan ininterrumpidamente.

Para obtener más información sobre todos los ajustes predeterminados que puede configurar al iniciar una actualización de instancias, consulte Comprensión de los valores predeterminados de una actualización de instancias.

Periodo de gracia de la comprobación de estado

Amazon EC2 Auto Scaling determina si una instancia está en buen estado en función de las comprobaciones de estado que utiliza su grupo de escalado automático. Para obtener más información, consulte Comprobaciones de estado para instancias en un grupo de escalado automático.

Para asegurarse de que estas comprobaciones de estado comiencen lo antes posible, no establezca demasiado alto el periodo de gracia de las comprobaciones de estado del grupo, sino lo suficientemente alto como para que las comprobaciones de estado de Elastic Load Balancing puedan determinar si hay un objetivo disponible para gestionar las solicitudes. Para obtener más información, consulte Establezca el periodo de gracia de la comprobación de estado para un grupo de escalado automático.

Compatibilidad de los tipos de instancias

Antes de cambiar el tipo de instancia, es recomendable comprobar que funciona con la plantilla de lanzamiento. Esto confirma la compatibilidad con la AMI que especificó. Por ejemplo, si lanzó las instancias originales desde una AMI paravirtual (PV), pero quiere cambiar a un tipo de instancia de generación actual que solo es compatible con una AMI de máquina virtual de hardware (HVM). En este caso, debe utilizar una AMI de HVM en la plantilla de lanzamiento.

Para confirmar la compatibilidad del tipo de instancia sin lanzar instancias, utilice el comando run-instances con la opción `--dry-run`, como se muestra en el siguiente ejemplo.

```
aws ec2 run-instances --launch-template LaunchTemplateName=my-template,Version='1' --
dry-run
```

Para obtener más información sobre cómo se determina la compatibilidad, consulte Compatibilidad para cambiar el tipo de instancia en la Guía del usuario de Amazon EC2.

Limitaciones

- Duración total: el periodo de tiempo máximo durante el cual una actualización de una instancia puede seguir reemplazando activamente instancias es de 14 días.

- Diferencia en el comportamiento específico de los grupos ponderados: si se configura un grupo de instancias mixtas con una ponderación de instancia mayor o igual a la capacidad deseada del grupo, Amazon EC2 Auto Scaling puede reemplazar todas las instancias con el estado `InService` a la vez. Para evitar esta situación, siga las recomendaciones del tema Configuración de un grupo de escalado automático para utilizar ponderación de instancias. Especifique una capacidad deseada que sea mayor que su ponderación máxima cuando utilice ponderaciones con su grupo de escalado automático.
- Tiempo de espera de una hora: cuando una actualización de instancias no puede continuar haciendo reemplazos porque está esperando reemplazar instancias en espera o protegidas contra la reducción horizontal, o las nuevas instancias no superan las comprobaciones de estado, Amazon EC2 Auto Scaling seguirá intentándolo durante una hora. También proporciona un mensaje de estado para ayudarlo a resolver el problema. Si el problema persiste después de una hora, la operación produce un error. La intención es darle tiempo para recuperarse si se trata de un problema temporal.
- Implementación del código a través de los datos del usuario: la opción Omitir coincidencia no comprueba los cambios de código que se implementan desde un script de datos de usuario. Si utiliza datos de usuario para extraer código nuevo e instalar estas actualizaciones en instancias nuevas, le recomendamos que desactive la opción de omisión de coincidencias para asegurarse de que todas las instancias reciban el código más reciente, incluso sin haber realizado una actualización de la versión de la plantilla de lanzamiento.
- Restricción de actualización: si intenta actualizar la plantilla de lanzamiento, la configuración de lanzamiento o la política de instancias mixtas de un grupo de escalado automático mientras está activa una actualización de instancias con la configuración deseada, la solicitud fallará y mostrará el siguiente error de validación: `An active instance refresh with a desired configuration exists. All configuration options derived from the desired configuration are not available for update while the instance refresh is active.`
- Selección de instancias basada en atributos: si su grupo de escalado automático utiliza la selección de instancias basada en atributos (al especificar `InstanceRequirements` en una política de instancias mixtas), no se admitirán los siguientes parámetros de actualización de instancias:
  - SkipMatching- La actualización de una instancia fallará si la inicia con el `SkipMatching` parámetro en un grupo de Auto Scaling que se haya `InstanceRequirements` configurado.

- DesiredConfiguration- La actualización de una instancia fallará si la inicia con el `DesiredConfiguration` parámetro en un grupo de Auto Scaling que se haya `InstanceRequirements` configurado.

  Si necesita realizar una actualización de instancias en un grupo de escalado automático con una selección de instancias basada en atributos, debe iniciar la actualización de instancias sin estos parámetros.
- La sustitución del volumen raíz no es compatible con Elastic Load Balancing.

Comprensión de los valores predeterminados de una actualización de instancias

Antes de iniciar una actualización de instancias, puede personalizar las diversas preferencias que afectan a la actualización de instancias. Algunos valores predeterminados de las preferencias varían en función de si utiliza la consola o la línea de comandos (AWS CLI o el AWS SDK).

En esta tabla se muestran los valores predeterminados de la configuración de actualización de instancias.

Opción	AWS CLI o SDK AWS	Consola de Amazon EC2 Auto Scaling
CloudWatch alarma	Deshabilitado (nulo)	Deshabilitado
Reversión automática	Deshabilitado (`false`)	Deshabilitado
Hora de hornear	Cero	Cero
Puntos de control	Deshabilitado (`false`)	Deshabilitado
Retraso de punto de comprobación	1 hora (`3600` segundos)	1 hora
Preparación de las instancias	La preparación de la instancia predeterminada, si está definida, o el periodo de gracia de la comprobación de estado, en caso contrario.	La preparación de la instancia predeterminada, si está definida, o el periodo de gracia de la comprobación de estado, en caso contrario.

Opción	AWS CLI o SDK AWS	Consola de Amazon EC2 Auto Scaling
Porcentaje máximo en buen estado	Varía en función de la política de mantenimiento de instancias. Si no hay una política de mantenimiento de instancias, el valor predeterminado es del 100 por ciento (nula).	Varía en función de la política de mantenimiento de instancias. Si no hay una política de mantenimiento de instancias, el valor predeterminado es del 100 por ciento (nula).
Porcentaje de buen estado mínimo	Varía en función de la política de mantenimiento de instancias. Si no hay una política de mantenimiento de instancias, el valor predeterminado es del 90 por ciento.	Varía en función de la política de mantenimiento de instancias. Si no hay una política de mantenimiento de instancias, el valor predeterminado es del 90 por ciento.
Instancias con protección de reducción horizontal	Wait	Ignore
Omisión de coincidencias	Deshabilitado (`false`)	Habilitado
Instancias en espera	Wait	Ignore

A continuación, se muestra una descripción de cada configuración:

CloudWatch alarma (**`AlarmSpecification`**)

La especificación CloudWatch de la alarma. CloudWatch las alarmas se pueden utilizar para identificar cualquier problema y hacer que la operación falle si se activa una `ALARM` alarma. Para obtener más información, consulte Inicio de una actualización de instancias con reversión automática.

Reversión automática (**`AutoRollback`**)

Controla si Amazon EC2 Auto Scaling revierte el grupo de escalado automático a su configuración anterior si se produce un error en la actualización de instancias. Para obtener más información, consulte Inversión de cambios con una reversión manual o automática.

Tiempo de incorporación (**`BakeTime`**)

Tiempo que se debe esperar al final de una actualización de instancias antes de que la actualización se considere completa.

Puntos de comprobación (**`CheckpointPercentages`**)

Controla si Amazon EC2 Auto Scaling reemplaza las instancias en fases. Esto resulta útil si necesita realizar verificaciones en las instancias antes de reemplazar todas las instancias. Para obtener más información, consulte Agregar puntos de control a una actualización de instancias.

Retraso de punto de comprobación (**`CheckpointDelay`**)

La cantidad de tiempo, en segundos, que debe esperar después de alcanzar un punto de comprobación antes de continuar. Para obtener más información, consulte Agregar puntos de control a una actualización de instancias.

Preparación de la instancia (**`InstanceWarmup`**)

Periodo de tiempo, en segundos, durante el cual Amazon EC2 Auto Scaling espera hasta que una instancia nueva se considere que una instancia nueva ha terminado de inicializarse antes de reemplazar la siguiente instancia. Si ya ha definido correctamente la preparación predeterminada de instancias del grupo de escalado automático, no necesita cambiarla (a menos que quiera anular la configuración predeterminada). Para obtener más información, consulte Establecimiento de la preparación predeterminada de instancias para un grupo de escalado automático.

Porcentaje máximo en buen estado (**`MaxHealthyPercentage`**)

El porcentaje de la capacidad deseada del grupo de escalado automático al cual el grupo puede aumentar al reemplazar instancias.

Porcentaje mínimo en buen estado (**`MinHealthyPercentage`**)

El porcentaje de la capacidad deseada del grupo de escalado automático que debe estar en servicio, en buen estado y listo para usarse antes de que la operación pueda continuar.

Instancias con protección de reducción horizontal (**`ScaleInProtectedInstances`**)

Controla que Amazon EC2 Auto Scaling encuentre instancias que estén protegidas contra la reducción horizontal. Para obtener más información sobre estas instancias, consulte Uso de la protección frente a la reducción horizontal de instancias para controlar la terminación de instancias.

Amazon EC2 Auto Scaling ofrece las siguientes opciones:

- Reemplazar (`Refresh`): reemplaza las instancias que están protegidas contra la reducción horizontal.
- Ignorar (`Ignore`): ignora las instancias que están protegidas contra la reducción horizontal y continúa reemplazando las instancias que no están protegidas.
- Esperar (`Wait`): espera una hora para que elimine la protección contra la reducción horizontal. Si no lo hace, se producirá un error en la actualización de instancias.

Omitir coincidencia (**`SkipMatching`**)

Controla si Amazon EC2 Auto Scaling omite el reemplazo de las instancias que coinciden con la configuración deseada. Si no se especifica ninguna configuración deseada, se omite el reemplazo de las instancias que tienen la misma plantilla de lanzamiento y los mismos tipos de instancias que el grupo de escalado automático utilizaba antes de que se iniciara la actualización de instancias. Para obtener más información, consulte Uso de una actualización de instancias con la omisión de coincidencias.

Instancias en espera (**`StandbyInstances`**)

Controla qué hace Amazon EC2 Auto Scaling si encuentra instancias en estado `Standby`. Para obtener más información sobre estas instancias, consulte Eliminación temporal de las instancias de un grupo de escalado automático.

Amazon EC2 Auto Scaling ofrece las siguientes opciones:

- Finalizar (`Terminate`): finaliza las instancias que se encuentran en estado `Standby`.
- Ignorar (`Ignore`): ignora las instancias que se encuentran en estado `Standby` y continúa reemplazando las instancias que están en estado `InService`.
- Esperar (`Wait`): espera una hora para que devuelva las instancias al servicio. Si no lo hace, se producirá un error en la actualización de instancias.

Inicie la actualización de una instancia con Consola de administración de AWS o AWS CLI

Important

Puede revertir una actualización de instancias que está en curso para anular los cambios. Para que esto funcione, el grupo de escalado automático tiene que cumplir los requisitos previos para utilizar las reversiones antes de iniciar la actualización de instancias. Para

obtener más información, consulte Inversión de cambios con una reversión manual o automática.

Los siguientes procedimientos te ayudan a iniciar una actualización de instancias con la tecla Consola de administración de AWS o AWS CLI.

Inicio de una actualización de instancias (consola)

Si es la primera vez que inicia una actualización de instancias, el uso de la consola lo ayudará a comprender las características y las opciones disponibles.

Inicio de una actualización de instancias en la consola (procedimiento básico)

Utilice el siguiente procedimiento si no definió previamente una política de instancias mixtas para el grupo de Auto Scaling. Si definió previamente una política de instancias mixtas, consulte Inicio de una actualización de instancias en la consola (grupo de instancias mixtas) para iniciar una actualización de instancias.

Para comenzar una actualización de instancias

1. Abra la consola Amazon EC2 en https://console.aws.amazon.com/ec2/y seleccione Auto Scaling Groups en el panel de navegación.
2. Seleccione la casilla situada junto al grupo de escalado automático.

   Se abre un panel dividido en la parte inferior de la página Auto Scaling groups (Grupos de Auto Scaling).
3. En la pestaña Instance refresh (Actualización de instancias) en Active Instance refresh (Actualización de instancias activas), elija Start instance refresh (Iniciar actualización de instancias).
4. En Configuración de disponibilidad, haga lo siguiente:

   a. Para el Método de reemplazo de instancias:

      - Si no ha establecido una política de mantenimiento de instancias en el grupo de escalado automático, la configuración predeterminada para el método de reemplazo de instancias es Finalice y lance. Este es el comportamiento predeterminado anterior de una actualización de instancias.

- Si establece una política de mantenimiento de instancias en el grupo de escalado automático, proporciona valores predeterminados para el método de reemplazo de instancias. Para anular la política de mantenimiento de instancias, seleccione Anular. La anulación solo se aplica a la actualización de instancias actual. La próxima vez que inicie una actualización de instancias, estos valores se restablecerán a los valores predeterminados de la política de mantenimiento de instancias.

En el siguiente procedimiento, se explica cómo actualizar el método de reemplazo de instancias.

i. Elija uno de los siguientes métodos de reemplazo de instancias:

   - Lance antes de terminar: primero se debe aprovisionar una nueva instancia antes de poder cancelar una instancia existente. Esta es una buena opción para las aplicaciones que prefieren la disponibilidad en lugar del ahorro de costos.
   - Finalice y lance: las instancias nuevas se aprovisionan al mismo tiempo que se terminan las instancias existentes. Esta es una buena opción para las aplicaciones que favorecen el ahorro de costos por encima de la disponibilidad. También es una buena opción para las aplicaciones que no deberían lanzar una capacidad superior a la disponible actualmente.
   - Comportamiento personalizado: esta opción permite configurar un rango mínimo y máximo personalizado para la cantidad de capacidad que quiere que esté disponible al reemplazar las instancias. Esto puede ayudarlo a lograr el equilibrio adecuado entre costo y disponibilidad.

ii. En Defina un porcentaje de buen estado, introduzca valores para uno o ambos de los siguientes campos. Los campos de activación varían según la opción que elija para el método de reemplazo de instancias.

   - Mínimo: establece el porcentaje de buen estado mínimo necesario para continuar con la actualización de instancias.
   - Máximo: establece el porcentaje máximo en buen estado posible durante la actualización de instancias.

iii. Amplíe la sección Ver la capacidad temporal estimada durante las sustituciones en función del tamaño del grupo actual para confirmar cómo se aplican los valores mínimo y máximo a su grupo. Los valores exactos utilizados dependen del valor de la capacidad deseada, que cambiará si el grupo escala.

iv. Amplíe la sección Definir un comportamiento alternativo para tamaños de reemplazo no válidos y, a continuación, elija si desea Infrinja el porcentaje máximo de buen estado para priorizar la disponibilidad o Infrinja el porcentaje mínimo de buen estado.

No se recomienda mantener la opción predeterminada de Infrinja el porcentaje mínimo de buen estado para grupos muy pequeños. Si solo hay una instancia en el grupo de escalado automático, iniciar una actualización de instancias puede provocar una interrupción.

Este paso configura el comportamiento alternativo si está utilizando un grupo de escalado automático que aún no tiene una política de mantenimiento de instancias. Esta opción no está disponible y no aparece cuando el grupo tiene una política de mantenimiento de instancias. Esta opción también solo está disponible para el método Finalice y lance. Otros métodos de reemplazo infringirán el porcentaje máximo en buen estado para priorizar la disponibilidad.

b. En Preparación de la instancia, ingrese el número de segundos desde que el estado de una nueva instancia cambia a `InService` hasta que finaliza su inicialización. Amazon EC2 Auto Scaling espera este tiempo antes de proceder a reemplazar la siguiente instancia.

Durante la preparación, una instancia recién lanzada tampoco se cuenta en las métricas de instancia agregadas del grupo de escalado automático (como `CPUUtilization`, `NetworkIn` y `NetworkOut`). Si ha agregado políticas de escalado al grupo de Auto Scaling, las actividades de escalado se ejecutan en paralelo. Si establece un intervalo largo para el periodo de preparación de la actualización de instancias, las instancias recién iniciadas tardan más tiempo en mostrarse en las métricas. Por lo tanto, un periodo de preparación adecuado evita que Amazon EC2 Auto Scaling realice el escalado sobre los datos de métricas obsoletos.

Si ya ha definido correctamente la preparación predeterminada de instancias del grupo de escalado automático, no necesita cambiarla. Sin embargo, si quiere anular el valor predeterminado, puede establecer un valor para esta opción. Para obtener más información sobre la preparación de las instancias predeterminada, consulte Establecimiento de la preparación predeterminada de instancias para un grupo de escalado automático.

5. En Configuración de actualización, haga lo siguiente:

a. (Opcional) En Checkpoints (Puntos de control), elija Enable checkpoints (Habilitar puntos de control) para reemplazar instancias mediante un enfoque progresivo o gradual de una actualización de instancias. Esto le ofrece tiempo adicional para la verificación entre

conjuntos de reemplazos. Si elige no habilitar los puntos de control, las instancias se reemplazan en una operación casi continua.

Si habilita los puntos de control, consulte Habilitar puntos de control (consola) para obtener pasos adicionales.

b. (Opcional) En Tiempo de incorporación, se especifica el tiempo que se debe esperar al final de la actualización de la instancia antes de que la actualización se considere completa.

c. Cómo habilitar o desactivar Skip matching (Omisión de coincidencias):

- Para omitir el reemplazo de las instancias que ya coinciden con la plantilla de lanzamiento, mantenga seleccionada la casilla de verificación Habilitar Omitir coincidencia.
- Si desactiva la casilla de verificación de omisión de coincidencias, se pueden reemplazar todas las instancias.

Cuando se habilita la omisión de coincidencias, puede configurar una nueva plantilla de lanzamiento o una nueva plantilla de lanzamiento, en lugar de utilizar la que ya existe. Hágalo en la sección Configuración deseada de la página Iniciar actualización de instancias.

> **Note**
>
> Para utilizar la característica de omisión de coincidencias y actualizar un grupo de Auto Scaling que actualmente utiliza una configuración de lanzamiento, debe seleccionar una plantilla de lanzamiento en Desired configuration (Configuración deseada). No se admite la omisión de coincidencias con una configuración de lanzamiento.

d. En Instancias en espera, seleccione Ignorar, Finalizar o Esperar. Esto determina lo que ocurre si se encuentran instancias en estado `Standby`. Para obtener más información, consulte Eliminación temporal de las instancias de un grupo de escalado automático.

Si elige Esperar, debe tomar medidas adicionales para devolver estas instancias al servicio. Si no lo hace, la actualización de instancias reemplazará a todas las instancias en estado `InService` y esperará una hora. A continuación, si queda alguna instancia en estado `Standby`, se producirá un error en la actualización de instancias. Para evitar esta situación, elija Ignorar o Finalizar estas instancias en su lugar.

e. En Instancias con protección de reducción horizontal, elija Ignorar, Reemplazar o Esperar. Esto determina lo que ocurre si se encuentran instancias protegidas contra la reducción horizontal. Para obtener más información, consulte Uso de la protección frente a la reducción horizontal de instancias para controlar la terminación de instancias.

   Si elige Esperar, debe tomar medidas adicionales para eliminar la protección contra la reducción horizontal de estas instancias. Si no lo hace, la actualización de instancias reemplazará a todas las instancias no protegidas y esperará una hora. A continuación, si queda alguna instancia protegida contra la reducción horizontal, se producirá un error en la actualización de instancias. Para evitar esta situación, elija Ignorar o Reemplazar estas instancias en su lugar.

6. (Opcional) Para la CloudWatch alarma, seleccione Activar CloudWatch alarmas y, a continuación, elija una o más alarmas. CloudWatch las alarmas se pueden utilizar para identificar cualquier problema y anular la operación si se activa una `ALARM` alarma. Para obtener más información, consulte Inicio de una actualización de instancias con reversión automática.

7. (Opcional) Amplíe la sección Configuración deseada para especificar las actualizaciones que desee realizar en el grupo de escalado automático.

   En este paso, puede elegir utilizar la sintaxis JSON o YAML para editar los valores de parámetros en lugar de realizar selecciones en la interfaz de la consola. Para ello, elija Use code editor (Usar editor de código) en lugar de Use console interface (Usar interfaz de consola). En el siguiente procedimiento, se explica cómo realizar selecciones con la interfaz de la consola.

   a. En Update launch template (Actualización de la plantilla de lanzamiento):

      - Si no creó una nueva plantilla de lanzamiento o una nueva versión de la plantilla de lanzamiento para su grupo de escalado automático, no seleccione esta casilla de verificación.
      - Si creó una nueva plantilla de lanzamiento o una nueva versión de la plantilla de lanzamiento, seleccione esta casilla de verificación. Al seleccionar esta opción, Amazon EC2 Auto Scaling le muestra la plantilla de lanzamiento actual y la versión actual de la plantilla de lanzamiento. También indica cualquier otra versión disponible. Elija la plantilla de lanzamiento y, a continuación, elija la versión.

        Después de elegir una versión, podrá ver la información de la versión. Esta es la versión de la plantilla de lanzamiento que se utilizará cuando se reemplacen instancias como parte de una actualización de instancias. Si la actualización de la instancia se realiza

correctamente, esta versión de la plantilla de lanzamiento también se utilizará cada vez que se lancen nuevas instancias, como en el escalado del grupo.

b. En Choose a set of instance types and purchase options to override the instance type in the launch template (Elegir un conjunto de instancias y opciones de compra para anular el tipo de instancia en la plantilla de lanzamiento):

- No seleccione esta casilla de verificación sin quiere utilizar el tipo de instancia y la opción de compra que especificó en la plantilla de lanzamiento.
- Active esta casilla de verificación si desea anular el tipo de instancia en la plantilla de lanzamiento o ejecutar instancias de spot. Puede agregar manualmente cada tipo de instancia o elegir un tipo de instancia principal y una opción de recomendación que recupere cualquier tipo de instancia coincidente adicional por usted. Si tiene previsto lanzar instancias de spot, le recomendamos que agregue algunos tipos de instancias diferentes. De esta forma, Amazon EC2 Auto Scaling puede lanzar otro tipo de instancia si no hay suficiente capacidad en las zonas de disponibilidad elegidas. Para obtener más información, consulte Grupos de Auto Scaling con varios tipos de instancia y opciones de compra.

> **Warning**
>
> No utilice instancias de spot con aplicaciones que no puedan gestionar una interrupción de las mismas. Se pueden producir interrupciones si el servicio de spot de Amazon EC2 necesita recuperar la capacidad.

Si activa esta casilla de verificación, asegúrese de que la plantilla de lanzamiento no solicite ya instancias de spot. No se puede utilizar una plantilla de lanzamiento que solicite instancias de spot para crear un grupo de escalado automático que utilice varios tipos de instancias e inicie instancias de spot y bajo demanda.

> **Note**
>
> Para configurar estas opciones en un grupo de Auto Scaling que actualmente utiliza una configuración de lanzamiento, debe seleccionar una plantilla de lanzamiento en Update launch template (Actualización de la plantilla de lanzamiento). No se admite la anulación del tipo de instancia en la configuración de lanzamiento.

8. (Opcional) En Configuración de la reversión, seleccione Habilitar la reversión automática para revertir automáticamente la actualización de instancias en caso de que se produzca un error.

   Esta configuración solo se puede habilitar cuando el grupo de escalado automático cumple con los requisitos previos para utilizar las reversiones.

   Para obtener más información, consulte Inversión de cambios con una reversión manual o automática.
9. Revise todas las selecciones para confirmar que todo está configurado correctamente.

   Una buena idea en este punto es verificar que las diferencias entre los cambios actuales y los propuestos no afecten a la aplicación de formas inesperadas o no deseadas. Para confirmar que el tipo de instancia es compatible con la plantilla de lanzamiento, consulte Compatibilidad de los tipos de instancias.
10. Cuando esté satisfecho con las selecciones de actualización de instancias, elija Iniciar actualización de instancias.

Inicio de una actualización de instancias en la consola (grupo de instancias mixtas)

Utilice el siguiente procedimiento si creó un grupo de Auto Scaling con una política de instancias mixtas. Si aún no definió una política de instancias mixtas para el grupo, consulte Inicio de una actualización de instancias en la consola (procedimiento básico) para iniciar una actualización de instancias.

Para comenzar una actualización de instancias

1. Abra la consola Amazon EC2 en https://console.aws.amazon.com/ec2/y seleccione Auto Scaling Groups en el panel de navegación.
2. Seleccione la casilla situada junto al grupo de escalado automático.

   Se abre un panel dividido en la parte inferior de la página Auto Scaling groups (Grupos de Auto Scaling).
3. En la pestaña Instance refresh (Actualización de instancias) en Active Instance refresh (Actualización de instancias activas), elija Start instance refresh (Iniciar actualización de instancias).
4. En Configuración de disponibilidad, haga lo siguiente:

   a. Para el Método de reemplazo de instancias:

- Si no ha establecido una política de mantenimiento de instancias en el grupo de escalado automático, la configuración predeterminada para el método de reemplazo de instancias es Finalice y lance. Este es el comportamiento predeterminado anterior de una actualización de instancias.
- Si establece una política de mantenimiento de instancias en el grupo de escalado automático, proporciona valores predeterminados para el método de reemplazo de instancias. Para anular la política de mantenimiento de instancias, seleccione Anular. La anulación solo se aplica a la actualización de instancias actual. La próxima vez que inicie una actualización de instancias, estos valores se restablecerán a los valores predeterminados de la política de mantenimiento de instancias.

En el siguiente procedimiento, se explica cómo actualizar el método de reemplazo de instancias.

i. Elija uno de los siguientes métodos de reemplazo de instancias:

   - Lance antes de terminar: primero se debe aprovisionar una nueva instancia antes de poder cancelar una instancia existente. Esta es una buena opción para las aplicaciones que prefieren la disponibilidad en lugar del ahorro de costos.
   - Finalice y lance: las instancias nuevas se aprovisionan al mismo tiempo que se terminan las instancias existentes. Esta es una buena opción para las aplicaciones que favorecen el ahorro de costos por encima de la disponibilidad. También es una buena opción para las aplicaciones que no deberían lanzar una capacidad superior a la disponible actualmente.
   - Comportamiento personalizado: esta opción permite configurar un rango mínimo y máximo personalizado para la cantidad de capacidad que quiere que esté disponible al reemplazar las instancias. Esto puede ayudarlo a lograr el equilibrio adecuado entre costo y disponibilidad.

ii. En Defina un porcentaje de buen estado, introduzca valores para uno o ambos de los siguientes campos. Los campos de activación varían según la opción que elija para el método de reemplazo de instancias.

   - Mínimo: establece el porcentaje de buen estado mínimo necesario para continuar con la actualización de instancias.

- Máximo: establece el porcentaje máximo en buen estado posible durante la actualización de instancias.

iii. Amplíe la sección Ver la capacidad temporal estimada durante las sustituciones en función del tamaño del grupo actual para confirmar cómo se aplican los valores mínimo y máximo a su grupo. Los valores exactos utilizados dependen del valor de la capacidad deseada, que cambiará si el grupo escala.

iv. Amplíe la sección Definir un comportamiento alternativo para tamaños de reemplazo no válidos y, a continuación, elija si desea Infrinja el porcentaje máximo de buen estado para priorizar la disponibilidad o Infrinja el porcentaje mínimo de buen estado.

No se recomienda mantener la opción predeterminada de Infrinja el porcentaje mínimo de buen estado para grupos muy pequeños. Si solo hay una instancia en el grupo de escalado automático, iniciar una actualización de instancias puede provocar una interrupción.

Este paso configura el comportamiento alternativo si está utilizando un grupo de escalado automático que aún no tiene una política de mantenimiento de instancias. Esta opción no está disponible y no aparece cuando el grupo tiene una política de mantenimiento de instancias. Esta opción también solo está disponible para el método Finalice y lance. Otros métodos de reemplazo infringirán el porcentaje máximo en buen estado para priorizar la disponibilidad.

b. En Preparación de la instancia, ingrese el número de segundos desde que el estado de una nueva instancia cambia a `InService` hasta que finaliza su inicialización. Amazon EC2 Auto Scaling espera este tiempo antes de proceder a reemplazar la siguiente instancia.

Durante la preparación, una instancia recién lanzada tampoco se cuenta en las métricas de instancia agregadas del grupo de escalado automático (como `CPUUtilization`, `NetworkIn` y `NetworkOut`). Si ha agregado políticas de escalado al grupo de Auto Scaling, las actividades de escalado se ejecutan en paralelo. Si establece un intervalo largo para el periodo de preparación de la actualización de instancias, las instancias recién iniciadas tardan más tiempo en mostrarse en las métricas. Por lo tanto, un periodo de preparación adecuado evita que Amazon EC2 Auto Scaling realice el escalado sobre los datos de métricas obsoletos.

Si ya ha definido correctamente la preparación predeterminada de instancias del grupo de escalado automático, no necesita cambiarla. Sin embargo, si quiere anular el valor predeterminado, puede establecer un valor para esta opción. Para obtener más información

sobre la preparación de las instancias predeterminada, consulte Establecimiento de la preparación predeterminada de instancias para un grupo de escalado automático.

5. En Configuración de actualización, haga lo siguiente:

   a. (Opcional) En Checkpoints (Puntos de control), elija Enable checkpoints (Habilitar puntos de control) para reemplazar instancias mediante un enfoque progresivo o gradual de una actualización de instancias. Esto le ofrece tiempo adicional para la verificación entre conjuntos de reemplazos. Si elige no habilitar los puntos de control, las instancias se reemplazan en una operación casi continua.

      Si habilita los puntos de control, consulte Habilitar puntos de control (consola) para obtener pasos adicionales.

   b. Cómo habilitar o desactivar Skip matching (Omisión de coincidencias):

      - Para omitir el reemplazo de instancias que ya coinciden con la plantilla de lanzamiento y cualquier anulación de tipo de instancia, mantenga seleccionada la casilla de verificación Habilitar Omitir coincidencia.
      - Si elige desactivar la casilla de verificación de omisión de coincidencias, se pueden reemplazar todas las instancias.

      Cuando se habilita la omisión de coincidencias, puede configurar una nueva plantilla de lanzamiento o una nueva plantilla de lanzamiento, en lugar de utilizar la que ya existe. Hágalo en la sección Configuración deseada de la página Iniciar actualización de instancias. También puede actualizar las anulaciones de tipo de instancia en Desired configuration (Configuración deseada).

   c. En Instancias en espera, seleccione Ignorar, Finalizar o Esperar. Esto determina lo que ocurre si se encuentran instancias en estado `Standby`. Para obtener más información, consulte Eliminación temporal de las instancias de un grupo de escalado automático.

      Si elige Esperar, debe tomar medidas adicionales para devolver estas instancias al servicio. Si no lo hace, la actualización de instancias reemplazará a todas las instancias en estado `InService` y esperará una hora. A continuación, si queda alguna instancia en estado `Standby`, se producirá un error en la actualización de instancias. Para evitar esta situación, elija Ignorar o Finalizar estas instancias en su lugar.

   d. En Instancias con protección de reducción horizontal, elija Ignorar, Reemplazar o Esperar. Esto determina lo que ocurre si se encuentran instancias protegidas contra la reducción

horizontal. Para obtener más información, consulte Uso de la protección frente a la reducción horizontal de instancias para controlar la terminación de instancias.

Si elige Esperar, debe tomar medidas adicionales para eliminar la protección contra la reducción horizontal de estas instancias. Si no lo hace, la actualización de instancias reemplazará a todas las instancias no protegidas y esperará una hora. A continuación, si queda alguna instancia protegida contra la reducción horizontal, se producirá un error en la actualización de instancias. Para evitar esta situación, elija Ignorar o Reemplazar estas instancias en su lugar.

6. (Opcional) Para la CloudWatch alarma, seleccione Activar CloudWatch alarmas y, a continuación, elija una o más alarmas. CloudWatch las alarmas se pueden utilizar para identificar cualquier problema y anular la operación si se activa una `ALARM` alarma. Para obtener más información, consulte Inicio de una actualización de instancias con reversión automática.
7. En la sección Desired configuration (Configuración deseada), haga lo siguiente:

   En este paso, puede elegir utilizar la sintaxis JSON o YAML para editar los valores de parámetros en lugar de realizar selecciones en la interfaz de la consola. Para ello, elija Use code editor (Usar editor de código) en lugar de Use console interface (Usar interfaz de consola). En el siguiente procedimiento, se explica cómo realizar selecciones con la interfaz de la consola.

   a. En Update launch template (Actualización de la plantilla de lanzamiento):

      - Si no creó una nueva plantilla de lanzamiento o una nueva versión de la plantilla de lanzamiento para su grupo de escalado automático, no seleccione esta casilla de verificación.
      - Si creó una nueva plantilla de lanzamiento o una nueva versión de la plantilla de lanzamiento, seleccione esta casilla de verificación. Al seleccionar esta opción, Amazon EC2 Auto Scaling le muestra la plantilla de lanzamiento actual y la versión actual de la plantilla de lanzamiento. También indica cualquier otra versión disponible. Elija la plantilla de lanzamiento y, a continuación, elija la versión.

        Después de elegir una versión, podrá ver la información de la versión. Esta es la versión de la plantilla de lanzamiento que se utilizará cuando se reemplacen instancias como parte de una actualización de instancias. Si la actualización de la instancia se realiza correctamente, esta versión de la plantilla de lanzamiento también se utilizará cada vez que se lancen nuevas instancias, como en el escalado del grupo.

b. En Use these settings to override the instance type and purchase option defined in the launch template (Utilizar esta configuración para anular el tipo de instancia y la opción de compra definidas en la plantilla de lanzamiento):

   Esta casilla de verificación está activada de forma predeterminada. Amazon EC2 Auto Scaling rellena cada parámetro con el valor que actualmente está configurado en la política de instancias mixtas para el grupo de Auto Scaling. Actualice solo los valores de los parámetros que desea cambiar. Para obtener orientación sobre esta configuración, consulte Grupos de Auto Scaling con varios tipos de instancia y opciones de compra.

   > **Warning**
   >
   > Le recomendamos que no desactive esta casilla de verificación. Desactívela solo si desea dejar de utilizar una política de instancias mixtas. Después de que la actualización de instancias se realiza correctamente, Amazon EC2 Auto Scaling actualiza el grupo para que coincida con la Desired configuration (Configuración deseada). Si ya no incluye una política de instancias mixtas, Amazon EC2 Auto Scaling termina gradualmente cualquier instancia de spot que se esté ejecutando actualmente y la reemplaza por instancias bajo demanda. O bien, si la plantilla de lanzamiento solicita instancias de spot, Amazon EC2 Auto Scaling termina gradualmente cualquier instancia bajo demanda que se esté ejecutando actualmente y la reemplaza por instancias de spot.

8. (Opcional) En Configuración de la reversión, seleccione Habilitar la reversión automática para revertir automáticamente la actualización de instancias en caso de que se produzca un error, independientemente del motivo.

   Esta configuración solo se puede habilitar cuando el grupo de escalado automático cumple con los requisitos previos para utilizar las reversiones.

   Para obtener más información, consulte Inversión de cambios con una reversión manual o automática.

9. Revise todas las selecciones para confirmar que todo está configurado correctamente.

   Una buena idea en este punto es verificar que las diferencias entre los cambios actuales y los propuestos no afecten a la aplicación de formas inesperadas o no deseadas. Para confirmar que el tipo de instancia es compatible con la plantilla de lanzamiento, consulte Compatibilidad de los tipos de instancias.

Cuando esté satisfecho con las selecciones de actualización de instancias, elija Iniciar actualización de instancias.

Inicio de una actualización de instancias (AWS CLI)

Para comenzar una actualización de instancias

Usa el siguiente start-instance-refreshcomando para iniciar una actualización de instancias desde AWS CLI. Puede especificar las preferencias que desee cambiar en un archivo de configuración JSON. Cuando haga referencia al archivo de configuración, proporcione la ruta y el nombre del archivo como se muestra en el ejemplo siguiente.

```
aws autoscaling start-instance-refresh --cli-input-json file://config.json
```

Contenido de `config.json`:

```
{
    "AutoScalingGroupName": "my-asg",
    "Preferences": {
      "InstanceWarmup": 60,
      "MinHealthyPercentage": 50,
      "AutoRollback": true,
      "ScaleInProtectedInstances": Ignore,
      "StandbyInstances": Terminate
    }
}
```

Si no se proporcionan preferencias, se usan los valores predeterminados. Para obtener más información, consulte Comprensión de los valores predeterminados de una actualización de instancias.

Ejemplo de código de salida:

```
{
    "InstanceRefreshId": "08b91cf7-8fa6-48af-b6a6-d227f40f1b9b"
}
```

Supervise la actualización de una instancia mediante Consola de administración de AWS o AWS CLI

Puedes supervisar una actualización de instancia en curso o consultar el estado de las actualizaciones de instancias anteriores de las últimas seis semanas con la tecla Consola de administración de AWS o AWS CLI.

Supervisión y comprobación del estado de una actualización de instancias

Para supervisar y comprobar el estado de la actualización de una instancia, use uno de los siguientes métodos:

Console

> **Tip**
>
> En este procedimiento, ya deberían aparecer las columnas con nombre. Para mostrar las columnas ocultas o cambiar el número de filas que aparecen, elija el ícono de engranaje en la esquina superior derecha de la sección para abrir el modal de preferencias. Actualice la configuración según sea necesario y seleccione Confirmar.

Para supervisar y comprobar el estado de una actualización de instancias (consola)

1. Abra la consola Amazon EC2 en https://console.aws.amazon.com/ec2/y seleccione Auto Scaling Groups en el panel de navegación.
2. Seleccione la casilla situada junto al grupo de escalado automático.

   Se abre un panel dividido en la parte inferior de la página.
3. En la pestaña Instance refresh (Actualización de instancias), en Instance refresh history (Historial de actualizaciones de instancias), puede determinar el estado de su solicitud en la columna Status (Estado). La operación entra en estado `Pending` mientras se está inicializando. A continuación, el estado debería cambiar rápidamente a `InProgress`. Cuando se actualizan todas las instancias, el estado cambia a `Successful`.
4. Para seguir supervisando el éxito o el fracaso de las actividades en curso, consulte las actividades de escalado del grupo. En la pestaña Activity (Actividad) en Activity history (Historial de actividades), cuando se lanza la actualización de instancias, verá entradas

cuando se terminan las instancias y otro conjunto de entradas cuando se lanzan las instancias. Si tiene numerosas actividades de escalado, puede ver más si elige el ícono > en la parte superior del historial de actividades. Para obtener información sobre la solución de problemas que podrían provocar el error de las actividades, consulte Solución de problemas de Amazon EC2 Auto Scaling.

5. (Opcional) En la pestaña Administración de instancias, en Instancias, puede revisar el progreso de instancias específicas según sea necesario.

AWS CLI

Cómo supervisar y comprobar el estado de una actualización de instancias (AWS CLI)

Use el siguiente comando describe-instance-refreshes.

```
aws autoscaling describe-instance-refreshes --auto-scaling-group-name my-asg
```

A continuación, se muestra un ejemplo del resultado.

Las actualizaciones de las instancias se ordenan por hora de inicio. En primer lugar, se describen las actualizaciones de instancias que aún están en curso.

```
{
  "InstanceRefreshes":[
    {
      "InstanceRefreshId":"08b91cf7-8fa6-48af-b6a6-d227f40f1b9b",
      "AutoScalingGroupName":"my-asg",
      "Status":"InProgress",
      "StatusReason":"Waiting for instances to warm up before continuing. For
 example: i-0645704820a8e83ff is warming up.",
      "StartTime":"2023-11-24T16:46:52+00:00",
      "PercentageComplete":50,
      "InstancesToUpdate":0,
      "Preferences":{
        "MaxHealthyPercentage":120,
        "MinHealthyPercentage":90,
        "InstanceWarmup":60,
        "SkipMatching":false,
        "AutoRollback":true,
        "ScaleInProtectedInstances":"Ignore",
        "StandbyInstances":"Ignore"
      }
```

```
    },
    {
      "InstanceRefreshId":"0e151305-1e57-4a32-a256-1fd14157c5ec",
      "AutoScalingGroupName":"my-asg",
      "Status":"Successful",
      "StartTime":"2023-11-22T13:53:37+00:00",
      "EndTime":"2023-11-22T13:59:45+00:00",
      "PercentageComplete":100,
      "InstancesToUpdate":0,
      "Preferences":{
        "MaxHealthyPercentage":120,
        "MinHealthyPercentage":90,
        "InstanceWarmup":60,
        "SkipMatching":false,
        "AutoRollback":true,
        "ScaleInProtectedInstances":"Ignore",
        "StandbyInstances":"Ignore"
      }
    }
  ]
}
```


Para seguir supervisando el éxito o el fracaso de las actividades en curso, consulte las actividades de escalado del grupo. Las actividades de escalado también le permiten obtener más detalles que le ayuden a solucionar problemas relacionados con la actualización de una instancia. Para obtener más información, consulte Solución de problemas de Amazon EC2 Auto Scaling.

Estados de actualización de instancias

Al iniciar una actualización de instancias, esta pasa al estado Pending. Pasa de Pendiente a InProgresshasta que llega a Exitoso, Fallado RollbackSuccessful, Cancelado o RollbackFailed.

Una actualización de instancias puede tener los siguientes estados:

Status	Description (Descripción)
Pendiente	Se creó la solicitud, pero la actualización de instancias no se ha iniciado.
InProgress	Hay una actualización de instancias en curso.
Successful	Se ha completado correctamente una actualización de instancias.

Status	Description (Descripción)
Con error	No se ha podido completar la actualización de instancias. Puede solucionar problemas utilizando el motivo del estado y las actividades de escalado.
Cancelling	Se está cancelando una actualización de instancias en curso.
Cancelled	Se ha cancelado la actualización de instancias.
RollbackInProgress	La actualización de instancias tiene un proceso de reversión en curso.
RollbackFailed	No se ha podido completar la reversión. Puede solucionar problemas utilizando el motivo del estado y las actividades de escalado.
RollbackSuccessful	Le reversión se ha completado correctamente.
Incorporación	Tiempo de incorporación de espera especificado después de que la actualización de una instancia haya terminado de actualizar las instancia s.

Reemplace los volúmenes raíz durante la actualización de la instancia

Contenido

- Funcionamiento
- Requisitos
- Inicia una actualización de instancias con Replace Root Volume
- Limitaciones
- Utilice enlaces de ciclo de vida para reemplazar el volumen raíz

Funcionamiento

La sustitución del volumen raíz actualiza las instancias sustituyendo únicamente el volumen raíz de EBS y, al mismo tiempo, mantiene la instancia en ejecución. Esto elimina la necesidad de lanzar nuevas instancias y evita posibles restricciones de capacidad. Este proceso preserva todos los demás recursos de la instancia, incluidos:

- Interfaces de red y direcciones IP
- Volúmenes de EBS que no son raíz
- Volúmenes y datos del almacén de instancias
- Grupos de seguridad y funciones de IAM

Durante la sustitución del volumen raíz, la aplicación sigue ejecutándose en la instancia existente. El volumen raíz original se separa, se crea un nuevo volumen raíz a partir de la AMI especificada y, a continuación, se adjunta a la misma instancia. Los volúmenes raíz originales se eliminan automáticamente después de reemplazarlos correctamente.

Cuando utilizas la estrategia de reemplazar el volumen raíz, las instancias pasan por los siguientes estados del ciclo de vida:

1. `ReplacingRootVolume`- Comienza el reemplazo del volumen raíz
2. `ReplacingRootVolume:Wait`- Espera a que finalice el ciclo de vida del enlace (si está configurado)
3. `ReplacingRootVolume:Proceed`- Procede con el reemplazo
4. `RootVolumeReplaced`- El reemplazo se completó correctamente

Una vez completada la sustitución del volumen raíz, Auto Scaling realiza comprobaciones de estado en las instancias actualizadas. Si se produce un error al reemplazar el volumen raíz de una instancia, Auto Scaling marca esa instancia para su finalización y la reemplaza por una nueva instancia.

Requisitos

- Su grupo de Auto Scaling debe usar una política de instancias mixtas
- Todas las anulaciones de la política de instancias mixtas deben especificar una `ImageId`
- AMIs debe contener solo un volumen raíz
- Todas las instancias deben coincidir con la configuración de la plantilla de lanzamiento del grupo
- Debe iniciar la actualización de la instancia con la configuración deseada que tenga una política de instancias mixtas que contenga `ImageId` anulaciones.

Inicia una actualización de instancias con Replace Root Volume

Para iniciar una actualización de instancias con Replace Root Volume (AWS CLI)

Usa el siguiente start-instance-refreshcomando para iniciar una actualización de instancias desde AWS CLI. Puede especificar las preferencias que desee cambiar en un archivo de configuración JSON. Cuando haga referencia al archivo de configuración, proporcione la ruta y el nombre del archivo como se muestra en el ejemplo siguiente.

```
aws autoscaling start-instance-refresh --cli-input-json file://config.json
```

Contenido de `config.json`:

Example

```
{
  "AutoScalingGroupName": "my-asg",
  "Strategy" : "ReplaceRootVolume",
  "DesiredConfiguration":{
    "MixedInstancesPolicy":{
      "LaunchTemplate": {
        "LaunchTemplateSpecification": {
          "LaunchTemplateId":"my-launch-template",
          "Version": "1"
        },
        "Overrides": [
          {
            "InstanceType":"c5.large",
            "ImageId":"ami-1234example"
          },
          {
            "InstanceType":"m5.large",
            "ImageId":"ami-2345example"
          }
        ]
      }
    }
  },
  "Preferences": {
    "InstanceWarmup": 60,
    "MinHealthyPercentage": 90,
    "AutoRollback": true,
    "ScaleInProtectedInstances": Ignore,
    "StandbyInstances": Ignore
  }
}
```

Si no se proporcionan preferencias, se usan los valores predeterminados. Para obtener más información, consulte Comprensión de los valores predeterminados de una actualización de instancias.

Ejemplo de código de salida:

```
{
    "InstanceRefreshId": "08b91cf7-8fa6-48af-b6a6-d227f40f1b9b"
}
```

Limitaciones

- No puede iniciar una actualización de instancias con Replace Root Volume si el grupo Auto Scaling de EC2 o la configuración de actualización de instancias deseada utilizan la versión de plantilla de lanzamiento `$Default` o `$Latest` la versión de actualización de instancias.
- No puede iniciar una actualización de instancias con Replace Root Volume en un grupo Amazon EC2 Auto Scaling si contiene instancias cuyo tipo de instancia no esté presente en la política de instancias mixtas.
- Durante una actualización de instancias con Replace Root Volume, solo puede adjuntar instancias con los tipos que estén presentes en la política de instancias mixtas de la configuración deseada.
- No puede iniciar una actualización de instancias con Replace Root Volume en un grupo de Auto Scaling Group de EC2 que tenga una piscina caliente.
- No puede añadir un pool caliente a un grupo de Amazon EC2 Auto Scaling que tenga una actualización de instancias activa con Replace Root Volume.

Utilice enlaces de ciclo de vida para reemplazar el volumen raíz

Las instancias que se someten a un reemplazo de volumen raíz siguen sus propias transiciones de ciclo de vida, lo que le permite invocar acciones (por ejemplo, una función Lambda) and/or antes y después del reemplazo. Para obtener más información sobre los estados del ciclo de vida durante el reemplazo del volumen raíz, consulte. Transiciones entre los estados del ciclo de vida de las instancias que están siendo reemplazadas por

Para obtener más información acerca de cómo agregar un enlace de ciclo de vida, consulte Incorporación de los enlaces de ciclo de vida en su grupo de escalado automático. Para obtener más

información sobre cómo completar una acción de ciclo de vida, consulte Cómo completar una acción del ciclo de vida en un grupo de escalado automático.

Antes de reemplazar el volumen raíz, un enlace durante el ciclo de vida puede resultar útil en los siguientes escenarios:

- Debes cerrar la aplicación correctamente antes de reemplazar el volumen raíz y reiniciar la instancia.
- Desea mover los datos del volumen raíz a otra ubicación antes de reemplazar el volumen.

Tras reemplazar el volumen raíz, un enlace durante el ciclo de vida puede resultar útil en los siguientes escenarios:

- Desea comprobar que la instancia está totalmente lista con la nueva AMI antes de que comience a recibir tráfico.
- Desea iniciar los datos de la aplicación que existían anteriormente en el volumen raíz.

Cuando agregue enlaces de ciclo de vida, tenga en cuenta lo siguiente:

- Cuando se configura un enlace de ciclo de vida para la acción `autoscaling:EC2_INSTANCE_TERMINATING` del ciclo de vida, una instancia a punto de ser sustituida por un volumen raíz se detiene para realizar una acción personalizada cuando alcanza el `ReplacingRootVolume:Wait` estado.
- Cuando se configura un enlace de ciclo de vida para la acción `autoscaling:EC2_INSTANCE_LAUNCHING` del ciclo de vida, una instancia a la que se le acaba de reemplazar su volumen raíz se detiene para realizar una acción personalizada cuando alcanza el `Pending:Wait` estado.

Cuando las instancias alcanzan un estado de espera, Amazon EC2 Auto Scaling envía una notificación. Las notificaciones de enlace del ciclo de vida de Auto Scaling contienen un `Action` campo con un valor `ReplaceRootVolume` que indica que la instancia se está sustituyendo por el volumen raíz.

En la EventBridge sección de esta guía encontrará ejemplos de estas notificaciones. Para obtener más información, consulte La actualización de instancias reemplaza los eventos del ciclo de vida del volumen raíz.

Cancela la actualización de una instancia mediante Consola de administración de AWS o AWS CLI

Puede cancelar una actualización de instancias que todavía está en curso. No puede cancelarla después de que haya terminado.

La cancelación de una actualización de instancias no revierte ninguna instancia que ya se haya reemplazado. Para revertir los cambios en las instancias, haga una reversión en su lugar. Para obtener más información, consulte Inversión de cambios con una reversión manual o automática.

Temas

- Cancelación de una actualización de instancias (consola)
- Cancelación de una actualización de instancias (AWS CLI)

Cancelación de una actualización de instancias (consola)

Cómo cancelar una actualización de instancias

1. Abra la consola Amazon EC2 en https://console.aws.amazon.com/ec2/y seleccione Auto Scaling Groups en el panel de navegación.
2. Seleccione la casilla situada junto al grupo de escalado automático.
3. En la pestaña Actualización de instancias en Actualización de instancias activa, elija Acciones, Cancelar.
4. Cuando deba confirmar la selección, haga clic en Confirm (Confirmar).

El estado de la actualización de instancias se establece en Cancelling. Una vez finalizada la cancelación, el estado de la actualización de instancias se establece en Cancelled.

Cancelación de una actualización de instancias (AWS CLI)

Cómo cancelar una actualización de instancias

Utilice el cancel-instance-refreshcomando de AWS CLI y proporcione el nombre del grupo de Auto Scaling.

```
aws autoscaling cancel-instance-refresh --auto-scaling-group-name my-asg
```

Ejemplo de código de salida:

```
{
    "InstanceRefreshId": "08b91cf7-8fa6-48af-b6a6-d227f40f1b9b"
}
```

Cómo cancelar una actualización de instancias sin esperar a que se produzca la transición de instancias

Utilice la `--no-wait-for-transitioning-instances` opción con el cancel-instance-refreshcomando de AWS CLI para cancelar la actualización de una instancia sin tener que esperar a que las instancias estén en transición.

```
aws autoscaling cancel-instance-refresh –auto-scaling-group-name my-asg --no-wait-for-transitioning-instances
```

Por defecto, cuando cancela una actualización de instancias, Amazon EC2 Auto Scaling espera a que se completen todos los lanzamientos y terminaciones en curso antes de establecer el estado en Cancelado. Si tiene instancias con periodos de inactividad prolongados o necesita iniciar una nueva actualización de instancias inmediatamente, puede usar el comando para no tener que esperar a que se produzcan estas operaciones de transición.

Con esta opción, el estado de actualización de la instancia cambia a Cancelado, lo que permite iniciar una nueva actualización de la instancia. Todos los lanzamientos y terminaciones en curso continúan funcionando en segundo plano.

Inversión de cambios con una reversión manual o automática

Puede revertir una actualización de instancias que todavía está en curso. No puede revertirla después de que haya terminado. Sin embargo, puede volver a actualizar el grupo de escalado automático si inicia una nueva actualización de instancias.

Al hacer la reversión, Amazon EC2 Auto Scaling reemplaza las instancias que se han implementado hasta ahora. Las nuevas instancias coinciden con la configuración que guardó por última vez en el grupo de escalado automático antes de iniciar la actualización de instancias.

Amazon EC2 Auto Scaling ofrece las siguientes formas de revertir cambios:

- Reversión manual: se inicia una reversión manualmente para invertir lo que se implementó hasta el punto de reversión.

- Reversión automática: Amazon EC2 Auto Scaling revierte automáticamente lo que se implementó si la actualización de la instancia falla por algún motivo o si CloudWatch alguna de las alarmas que especifique entra en ese estado. `ALARM`

Contenido

- Consideraciones
- Inicio manual de una reversión
- Inicio de una actualización de instancias con reversión automática

Consideraciones

Las siguientes consideraciones se aplican cuando se utiliza una reversión:

- La opción de reversión solo está disponible si especifica la configuración deseada como parte del inicio de una actualización de instancias.
- Solo puede revertir a una versión anterior de una plantilla de lanzamiento si la versión es una versión numerada específica. La opción de reversión no está disponible si el grupo de escalado automático está configurado para usar la versión de plantilla `$Latest` o `$Default`.
- Tampoco puede volver a una plantilla de lanzamiento que esté configurada para usar un alias de AMI del almacén de AWS Systems Manager parámetros.
- La configuración que guardó por última vez en el grupo de escalado automático debe encontrarse en un estado estable. Si no está en un estado estable, el flujo de trabajo de reversión seguirá ejecutándose, pero al final se producirá un error. Hasta que no se resuelva el problema, es posible que el grupo de escalado automático se encuentre en un estado de error y ya no pueda lanzar instancias correctamente. Esto podría afectar a la disponibilidad del servicio o la aplicación.

Inicio manual de una reversión

Console

Inicio manual de una reversión de una actualización de instancias (consola)

1. Abra la consola Amazon EC2 en https://console.aws.amazon.com/ec2/y seleccione Auto Scaling Groups en el panel de navegación.
2. Seleccione la casilla situada junto al grupo de escalado automático.

3. En la pestaña Actualización de instancias en Actualización de instancias activa, elija Acciones, Iniciar reversión.
4. Cuando deba confirmar la selección, haga clic en Confirm (Confirmar).

AWS CLI

Inicio manual de una reversión de una actualización de instancias (AWS CLI)

Utilice el rollback-instance-refreshcomando de AWS CLI y proporcione el nombre del grupo de Auto Scaling.

```
aws autoscaling rollback-instance-refresh --auto-scaling-group-name my-asg
```

Ejemplo de código de salida:

```
{
    "InstanceRefreshId": "08b91cf7-8fa6-48af-b6a6-d227f40f1b9b"
}
```

> **Tip**
>
> Si este comando arroja un error, asegúrese de haber actualizado la versión AWS CLI local a la última versión.

Inicio de una actualización de instancias con reversión automática

Con la función de reversión automática, puedes revertir automáticamente la actualización de la instancia cuando se produce un error, por ejemplo, cuando hay errores o cuando se activa una CloudWatch alarma de Amazon específicaALARM.

Si habilita la reversión automática y se producen errores al reemplazar las instancias, la actualización de instancias intentará completar todas las sustituciones durante una hora antes de que se produzca un error y se revierta. Por lo general, estos errores se deben a errores en el lanzamiento de EC2, a una mala configuración de las comprobaciones de estado o a que no se ignoran ni permiten la finalización de las instancias que están en el estado `Standby` o protegidas contra la reducción horizontal.

La especificación de CloudWatch las alarmas es opcional. Para especificar una alarma, primero tiene que crearla. Puede especificar alarmas de métricas y alarmas compuestas. Para obtener información sobre cómo crear la alarma, consulta la Guía del CloudWatch usuario de Amazon. Si utiliza las métricas de Elastic Load Balancing como ejemplo, si utiliza un equilibrador de carga de aplicación, puede utilizar las métricas `HTTPCode_ELB_5XX_Count` y `HTTPCode_ELB_4XX_Count`.

Consideraciones

- Si especificas una CloudWatch alarma pero no habilitas la reversión automática y el estado de alarma pasa a`ALARM`, la actualización de la instancia fallará sin revertirla.
- Puede elegir un máximo de 10 alarmas al iniciar una actualización de instancias.
- Al elegir una CloudWatch alarma, la alarma debe estar en un estado compatible. Si el estado de la alarma es `INSUFFICIENT_DATA` o `ALARM`, recibirá un error al intentar iniciar la actualización de instancias.
- Al crear una alarma para que la utilice Amazon EC2 Auto Scaling, la alarma debe incluir cómo gestionar los puntos de datos faltantes. Si a una métrica le faltan puntos de datos por diseño, el estado de la alarma es `INSUFFICIENT_DATA` durante esos períodos. Cuando esto ocurre, Amazon EC2 Auto Scaling no puede instancias hasta que se encuentren nuevos puntos de datos. Para forzar la alarma a fin de mantener el estado anterior `ALARM` o `OK`, puede optar por ignorar los datos que faltan en su lugar. Para obtener más información, consulta Cómo configurar el modo en que las alarmas tratan los datos faltantes en la Guía del CloudWatch usuario de Amazon.

Console

Inicio de una actualización de instancias con reversión automática (consola)

1. Abra la consola Amazon EC2 en https://console.aws.amazon.com/ec2/y seleccione Auto Scaling Groups en el panel de navegación.
2. Seleccione la casilla situada junto al grupo de escalado automático.
3. En la pestaña Instance refresh (Actualización de instancias) en Active Instance refresh (Actualización de instancias activas), elija Start instance refresh (Iniciar actualización de instancias).
4. Siga el procedimiento Inicio de una actualización de instancias (consola) y configure su actualización de instancia según sea necesario.

5. (Opcional) En Actualizar la configuración, para la CloudWatch alarma, seleccione Activar CloudWatch alarmas y, a continuación, elija una o más alarmas para identificar cualquier problema y hacer que no funcione si se activa una alarma`ALARM`.
6. En Configuración de reversión, seleccione Habilitar la reversión automática para revertir automáticamente una actualización de instancias fallida a la configuración que guardó por última vez en el grupo de escalado automático antes de iniciar la actualización de instancias.
7. Revise sus selecciones y, a continuación, elija Iniciar la actualización de instancias.

AWS CLI

Inicio de una actualización de instancias con reversión automática (AWS CLI)

Utilice el start-instance-refreshcomando y especifique `true` la `AutoRollback` opción en`Preferences`.

En el siguiente ejemplo, se muestra cómo iniciar una actualización de instancias que se revertirá automáticamente si se produce algún error. Sustituya los valores del parámetro *`italicized`* por sus propios valores.

```
aws autoscaling start-instance-refresh --cli-input-json file://config.json
```

Contenido de `config.json`.

```
{
    "AutoScalingGroupName": "my-asg",
    "DesiredConfiguration": {
      "LaunchTemplate": {
          "LaunchTemplateName": "my-launch-template",
          "Version": "1"
       }
    },
    "Preferences": {
      "AutoRollback": true
    }
}
```

Como alternativa, para revertir automáticamente la actualización de la instancia cuando se produce un error en la actualización de la instancia o cuando una CloudWatch alarma específica

está en ese ALARM estado, especifique la AlarmSpecification opción Preferences y proporcione el nombre de la alarma, como en el siguiente ejemplo. Sustituya los valores del parámetro *italicized* por sus propios valores.

```
{
    "AutoScalingGroupName": "my-asg",
    "DesiredConfiguration": {
      "LaunchTemplate": {
          "LaunchTemplateName": "my-launch-template",
          "Version": "1"
       }
    },
    "Preferences": {
      "AutoRollback": true,
      "AlarmSpecification": { "Alarms": [ "my-alarm" ] }
    }
}
```

Si se ejecuta correctamente, el comando devolverá información similar a la siguiente.

```
{
  "InstanceRefreshId": "08b91cf7-8fa6-48af-b6a6-d227f40f1b9b"
}
```

> **Tip**
>
> Si este comando arroja un error, asegúrate de haber actualizado la versión AWS CLI local a la última versión.

Uso de una actualización de instancias con la omisión de coincidencias

La opción de omisión de coincidencias le dice a Amazon EC2 Auto Scaling que ignore las instancias que ya tienen sus actualizaciones más recientes. De esta forma, no reemplaza más instancias de las necesarias. Esto es útil cuando quiere asegurarse de que su grupo de escalado automático utilice una versión determinada de la plantilla de lanzamiento y solo sustituya las instancias que usan una versión diferente.

Tenga en cuenta las siguientes consideraciones para omitir coincidencias:

- Si comienza una actualización de instancias con la omisión de coincidencias y una configuración deseada, Amazon EC2 Auto Scaling comprueba si hay instancias que coincidan con su configuración deseada. A continuación, solo reemplaza las instancias que no coincidan con la configuración deseada. Cuando la actualización de instancias se lleva a cabo correctamente, Amazon EC2 Auto Scaling actualiza el grupo para que coincida con la configuración deseada.
- Si omite las coincidencias al iniciar una actualización de instancias, pero no especifica la configuración deseada, Amazon EC2 Auto Scaling comprobará si alguna instancia coincide con la configuración que guardó por última vez en el grupo de escalado automático. A continuación, solo reemplaza las instancias que no coincidan con la última configuración guardada.
- Puede utilizar la omisión de coincidencias con una nueva plantilla de lanzamiento, una nueva versión de la plantilla de lanzamiento actual o un conjunto de tipos de instancia. Si habilita la omisión de coincidencias, pero ninguna de estas opciones cambia, la actualización de instancias se ejecutará inmediatamente sin reemplazar ninguna instancia. Si ha efectuado otros cambios en la configuración deseada (como cambiar la estrategia de asignación de spot), Amazon EC2 Auto Scaling espera a que se realice correctamente la actualización de instancias. A continuación, actualiza la configuración del grupo de escalado automático para reflejar la nueva configuración deseada.
- No puede utilizar la omisión de coincidencias con una configuración de lanzamiento nueva.
- Al iniciar una actualización de instancias y proporcionar una configuración deseada, Amazon EC2 Auto Scaling se asegura de que todas las instancias utilicen la configuración deseada. Por lo tanto, si especifica `$Default` o `$Latest` como la versión deseada para su plantilla de lanzamiento y, a continuación, crea una nueva versión de la plantilla de lanzamiento mientras se está actualizando la instancia, todas las instancias que ya se hayan reemplazado volverán a sustituirse.
- La omisión de coincidencias no sabe si un script de datos de usuario de la plantilla de lanzamiento extraerá el código actualizado y lo instalará en nuevas instancias. Como resultado, es posible que, al omitir las coincidencias, no se reemplacen las instancias que tengan instalado un código desactualizado. En este caso, debe desactivar la omisión de coincidencias para asegurarse de que todas las instancias reciban el código más reciente, incluso sin actualizar la versión de la plantilla de inicialziación.

En esta sección se incluyen AWS CLI instrucciones para iniciar una actualización de instancias con la opción de omisión de coincidencias habilitada. Para obtener instrucciones sobre cómo utilizar la consola, consulte Inicio de una actualización de instancias (consola).

Omisión de coincidencias (procedimiento básico)

Siga los pasos de esta sección para utilizar el y AWS CLI hacer lo siguiente:

- Cree la plantilla de lanzamiento que quiera aplicar a las instancias.
- Comience una actualización de instancias para aplicar la plantilla de lanzamiento a un grupo de escalado automático. Si no habilita la omisión de coincidencias, se reemplazarán todas las instancias. Esto se aplica incluso si la plantilla de lanzamiento utilizada para aprovisionar la instancia es la misma que se especificó para la configuración deseada.

Uso de la omisión de coincidencias con una nueva plantilla de lanzamiento

1. Use el create-launch-templatecomando para crear una nueva plantilla de lanzamiento para su grupo de Auto Scaling. Incluya la opción `--launch-template-data` y la entrada JSON que define los detalles de las instancias que se crean para el grupo de escalado automático.

   Por ejemplo, utilice el siguiente comando para crear una plantilla de lanzamiento básica con el ID de AMI *ami-0123456789abcdef0* y el tipo de instancia *t2.micro*.

   ```
   aws ec2 create-launch-template --launch-template-name my-template-for-auto-scaling
    --version-description version1 \
     --launch-template-data
    '{"ImageId":"ami-0123456789abcdef0","InstanceType":"t2.micro"}'
   ```

   Si se ejecuta correctamente, el comando devolverá información similar a la siguiente.

   ```
   {
      "LaunchTemplate": {
        "LaunchTemplateId": "lt-068f72b729example",
        "LaunchTemplateName": "my-template-for-auto-scaling",
        "CreatedBy": "arn:aws:iam::123456789012:user/Bob",
        "CreateTime": "2023-01-30T18:16:06.000Z",
        "DefaultVersionNumber": 1,
        "LatestVersionNumber": 1
     }
   }
   ```

   Para obtener más información, consulte Ejemplos de creación y administración de plantillas de lanzamiento con AWS CLI.

2. Usa el start-instance-refreshcomando para iniciar el flujo de trabajo de reemplazo de instancias y aplica tu nueva plantilla de lanzamiento con el ID*lt-068f72b729example*. Como la plantilla de lanzamiento es nueva, solo tiene una versión. Esto significa que la versión 1 de la plantilla de lanzamiento es el objetivo de esta actualización de instancias. Si se produce un evento de escalado horizontal durante la actualización de instancias y Amazon EC2 Auto Scaling aprovisiona nuevas instancias con la versión 1 de esta plantilla de lanzamiento, no se reemplazarán. Cuando se finaliza adecuadamente la operación, la nueva plantilla de lanzamiento se aplica de manera correcta en un grupo de escalado automático.

```
aws autoscaling start-instance-refresh --cli-input-json file://config.json
```

Contenido de `config.json`.

```
{
    "AutoScalingGroupName": "my-asg",
    "DesiredConfiguration": {
      "LaunchTemplate": {
          "LaunchTemplateId": "lt-068f72b729example",
          "Version": "$Default"
       }
    },
    "Preferences": {
      "SkipMatching": true
    }
}
```

Si se ejecuta correctamente, el comando devolverá información similar a la siguiente.

```
{
  "InstanceRefreshId": "08b91cf7-8fa6-48af-b6a6-d227f40f1b9b"
}
```

Omisión de coincidencias (grupos de instancias mixtas)

Si tiene un grupo de Auto Scaling con una política de instancias mixtas, siga los pasos de esta sección AWS CLI para iniciar una actualización de instancias con la opción de omitir la coincidencia. Dispone de las opciones siguientes:

- Proporcione una nueva plantilla de lanzamiento para aplicarla a todos los tipos de instancias especificados en la política.
- Proporcione un conjunto actualizado de tipos de instancias cambiando o sin cambiar la plantilla de lanzamiento de la política. Por ejemplo, es posible que quiera alejarse de los tipos de instancias no deseados. Usaría la plantilla de lanzamiento tal como está, sin cambiar la AMI, los grupos de seguridad ni otros detalles de las instancias que se van a reemplazar.

Siga los pasos de una de las siguientes secciones, en función de la opción que se adapte a sus necesidades.

Uso de la omisión de coincidencias con una nueva plantilla de lanzamiento

1. Use el create-launch-templatecomando para crear una nueva plantilla de lanzamiento para su grupo de Auto Scaling. Incluya la opción `--launch-template-data` y la entrada JSON que define los detalles de las instancias que se crean para el grupo de escalado automático.

   Por ejemplo, utilice el siguiente comando para crear una plantilla de lanzamiento con el ID de AMI *ami-0123456789abcdef0*.

   ```
   aws ec2 create-launch-template --launch-template-name my-new-template --version-
   description version1 \
     --launch-template-data '{"ImageId":"ami-0123456789abcdef0"}'
   ```

   Si se ejecuta correctamente, el comando devolverá información similar a la siguiente.

   ```
   {
      "LaunchTemplate": {
        "LaunchTemplateId": "lt-04d5cc9b88example",
        "LaunchTemplateName": "my-new-template",
        "CreatedBy": "arn:aws:iam::123456789012:user/Bob",
        "CreateTime": "2023-01-31T15:56:02.000Z",
        "DefaultVersionNumber": 1,
        "LatestVersionNumber": 1
     }
   }
   ```

   Para obtener más información, consulte Ejemplos de creación y administración de plantillas de lanzamiento con AWS CLI.

2. Para ver la política de instancias mixtas existente para su grupo de Auto Scaling, ejecute el describe-auto-scaling-groupscomando. Necesitará esta información en el siguiente paso, cuando inicie la actualización de instancias.

   El siguiente comando de ejemplo devuelve la política de instancias mixtas configurada para el grupo de escalado automático denominado *my-asg*.

   ```
   aws autoscaling describe-auto-scaling-groups --auto-scaling-group-name my-asg
   ```

   Si se ejecuta correctamente, el comando devolverá información similar a la siguiente.

   ```
   {
     "AutoScalingGroups":[
       {
         "AutoScalingGroupName":"my-asg",
         "AutoScalingGroupARN":"arn",
         "MixedInstancesPolicy":{
           "LaunchTemplate":{
             "LaunchTemplateSpecification":{
               "LaunchTemplateId":"lt-073693ed27example",
               "LaunchTemplateName":"my-old-template",
               "Version":"$Default"
             },
             "Overrides":[
               {
                 "InstanceType":"c5.large"
               },
               {
                 "InstanceType":"c5a.large"
               },
               {
                 "InstanceType":"m5.large"
               },
               {
                 "InstanceType":"m5a.large"
               }
             ]
           },
           "InstancesDistribution":{
             "OnDemandAllocationStrategy":"prioritized",
             "OnDemandBaseCapacity":1,
             "OnDemandPercentageAboveBaseCapacity":50,
   ```

```
            "SpotAllocationStrategy":"price-capacity-optimized"
          }
        },
        "MinSize":1,
        "MaxSize":5,
        "DesiredCapacity":4,
        ...
      }
    ]
}
```

3. Usa el start-instance-refreshcomando para iniciar el flujo de trabajo de reemplazo de instancias y aplica tu nueva plantilla de lanzamiento con el ID*lt-04d5cc9b88example*. Como la plantilla de lanzamiento es nueva, solo tiene una versión. Esto significa que la versión 1 de la plantilla de lanzamiento es el objetivo de esta actualización de instancias. Si se produce un evento de escalado horizontal durante la actualización de instancias y Amazon EC2 Auto Scaling aprovisiona nuevas instancias con la versión 1 de esta plantilla de lanzamiento, no se reemplazarán. Cuando se finaliza adecuadamente la operación, la política de instancias mixtas se aplica de manera correcta en un grupo de escalado automático.

```
aws autoscaling start-instance-refresh --cli-input-json file://config.json
```

Contenido de `config.json`.

```
{
  "AutoScalingGroupName":"my-asg",
  "DesiredConfiguration":{
    "MixedInstancesPolicy":{
      "LaunchTemplate":{
        "LaunchTemplateSpecification":{
          "LaunchTemplateId":"lt-04d5cc9b88example",
          "Version":"$Default"
        },
        "Overrides":[
          {
            "InstanceType":"c5.large"
          },
          {
            "InstanceType":"c5a.large"
          },
          {
```

```
            "InstanceType":"m5.large"
          },
          {
            "InstanceType":"m5a.large"
          }
        ]
      },
      "InstancesDistribution":{
        "OnDemandAllocationStrategy":"prioritized",
        "OnDemandBaseCapacity":1,
        "OnDemandPercentageAboveBaseCapacity":50,
        "SpotAllocationStrategy":"price-capacity-optimized"
        }
      }
    }
  },
  "Preferences":{
    "SkipMatching":true
  }
}
```

Si se ejecuta correctamente, el comando devolverá información similar a la siguiente.

```
{
  "InstanceRefreshId": "08b91cf7-8fa6-48af-b6a6-d227f40f1b9b"
}
```

En el siguiente procedimiento, proporcionará un conjunto actualizado de tipos de instancias sin cambiar la plantilla de lanzamiento.

Para utilizar la omisión de coincidencias con un conjunto actualizado de tipos de instancias

1. Para ver la política de instancias mixtas existente para su grupo de Auto Scaling, ejecute el describe-auto-scaling-groupscomando. Necesitará esta información en el siguiente paso, cuando inicie la actualización de instancias.

   El siguiente comando de ejemplo devuelve la política de instancias mixtas configurada para el grupo de escalado automático denominado *my-asg*.

   ```
   aws autoscaling describe-auto-scaling-groups --auto-scaling-group-name my-asg
   ```

Si se ejecuta correctamente, el comando devolverá información similar a la siguiente.

```
{
  "AutoScalingGroups":[
    {
      "AutoScalingGroupName":"my-asg",
      "AutoScalingGroupARN":"arn",
      "MixedInstancesPolicy":{
        "LaunchTemplate":{
          "LaunchTemplateSpecification":{
            "LaunchTemplateId":"lt-073693ed27example",
            "LaunchTemplateName":"my-template-for-auto-scaling",
            "Version":"$Default"
          },
          "Overrides":[
            {
              "InstanceType":"c5.large"
            },
            {
              "InstanceType":"c5a.large"
            },
            {
              "InstanceType":"m5.large"
            },
            {
              "InstanceType":"m5a.large"
            }
          ]
        },
        "InstancesDistribution":{
          "OnDemandAllocationStrategy":"prioritized",
          "OnDemandBaseCapacity":1,
          "OnDemandPercentageAboveBaseCapacity":50,
          "SpotAllocationStrategy":"price-capacity-optimized"
        }
      },
      "MinSize":1,
      "MaxSize":5,
      "DesiredCapacity":4,
      ...
    }
  ]
```

```
}
```

2. Utilice el start-instance-refreshcomando para iniciar el flujo de trabajo de reemplazo de instancias y aplicar las actualizaciones. Si quiere reemplazar las instancias que utilizan tipos de instancia específicos, la configuración deseada debe especificar la política de instancias mixtas solo con los tipos de instancia deseados. Puede elegir si quiere agregar nuevos tipos de instancias en su lugar.

   El siguiente comando de ejemplo inicia una actualización de instancias sin el tipo de instancia no deseado *m5a.large*. Cuando un tipo de instancia del grupo no coincide con ninguno de los tres tipos de instancias, las instancias se reemplazan. (Tenga en cuenta que una actualización de instancias no elige los tipos de instancia desde los que aprovisionar las nuevas instancias; en su lugar, lo hacen las estrategias de asignación). Cuando se finaliza adecuadamente la operación, la política de instancias mixtas se aplica de manera correcta en un grupo de escalado automático.

   ```
   aws autoscaling start-instance-refresh --cli-input-json file://config.json
   ```

   Contenido de `config.json`

   ```
   {
     "AutoScalingGroupName":"my-asg",
     "DesiredConfiguration":{
       "MixedInstancesPolicy":{
         "LaunchTemplate":{
           "LaunchTemplateSpecification":{
             "LaunchTemplateId":"lt-073693ed27example",
             "Version":"$Default"
           },
           "Overrides":[
             {
               "InstanceType":"c5.large"
             },
             {
               "InstanceType":"c5a.large"
             },
             {
               "InstanceType":"m5.large"
             }
           ]
         },
   ```

```
      "InstancesDistribution":{
        "OnDemandAllocationStrategy":"prioritized",
        "OnDemandBaseCapacity":1,
        "OnDemandPercentageAboveBaseCapacity":50,
        "SpotAllocationStrategy":"price-capacity-optimized"
        }
      }
    }
  },
  "Preferences":{
    "SkipMatching":true
  }
}
```

Agregar puntos de control a una actualización de instancias

Al utilizar una actualización de instancias, puede elegir reemplazar instancias por fases, de modo que pueda hacer verificaciones en las instancias a medida que avanza. Para realizar un reemplazo por fases, agregue puntos de control, que son puntos en el tiempo en los que se detiene la actualización de instancias. El uso de puntos de control le ofrece un mayor control sobre cómo elige actualizar el grupo de Auto Scaling. Lo ayuda a confirmar que la aplicación va a funcionar de manera fiable y predecible.

Contenido

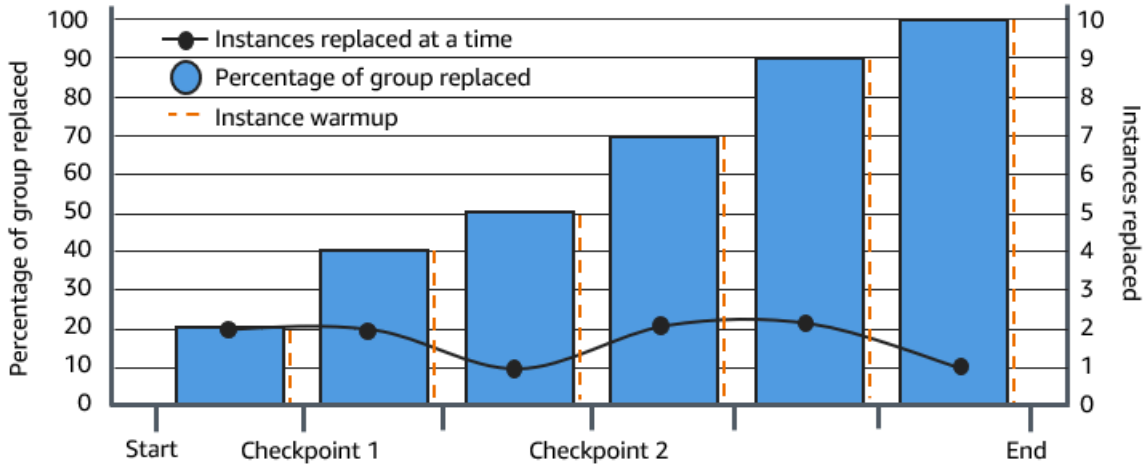
- Funcionamiento
- Consideraciones
- Habilite los puntos de control mediante el uso de o Consola de administración de AWS AWS CLI

Funcionamiento

Al comenzar una actualización de instancias, debe especificar los puntos de control como porcentajes del número total de instancias del grupo de escalado automático. Estos puntos de control indican el porcentaje mínimo de instancias del grupo de escalado automático que deben ser instancias nuevas antes de que se considere que se ha alcanzado el punto de control. Por ejemplo, si sus puntos de control son `[20, 50, 100]`, el primer punto de control se alcanza cuando el 20 por ciento de las instancias son nuevas; el segundo, cuando el 50 por ciento son nuevas; y el punto de control final, cuando todas las instancias son nuevas.

Amazon EC2 Auto Scaling ajusta el ritmo de los reemplazos de instancias para cumplir con los porcentajes de puntos de control especificados y, al mismo tiempo, mantiene el porcentaje de buen estado mínimo del grupo. Para alcanzar un porcentaje de puntos de control, Amazon EC2 Auto Scaling a veces reemplaza menos, pero nunca más, de lo que permite el porcentaje de buen estado mínimo.

Considere el siguiente grupo de escalado automático que tiene 10 instancias. Los porcentajes de punto de control son `[20,50,100]`; el porcentaje de buen estado mínimo es del 80 por ciento y el porcentaje máximo en buen estado es del 100 por ciento. Para mantener el porcentaje de buen estado mínimo, solo pueden reemplazarse dos instancias por vez. En el siguiente diagrama se resume el proceso de reemplazo de instancias antes de que se alcance un punto de control.



En el ejemplo anterior, hay un período de preparación de instancias para cada nueva instancia que se inicie. También puede que tenga un enlace de ciclo de vida que ponga una instancia en estado de espera y luego realice una acción personalizada durante el proceso de lanzamiento o terminación.

Amazon EC2 Auto Scaling emite eventos para cada punto de control, excepto para el 100 por ciento. Puede añadir una EventBridge regla para enviar los eventos a un destino como Amazon SNS. De esta forma, recibirá una notificación cuando pueda ejecutar las verificaciones necesarias. Para obtener más información, consulte Cree EventBridge reglas (por ejemplo, actualice los eventos).

Consideraciones

Tenga en cuenta las siguientes consideraciones al utilizar puntos de control:

- Dado que los puntos de control se basan en porcentajes, el número de instancias que se reemplazan cambia con el tamaño del grupo. Cuando se produce una actividad de escalado horizontal y aumenta el tamaño del grupo, una operación en curso podría volver a alcanzar un

punto de control. Si sucede esto, Amazon EC2 Auto Scaling envía otra notificación y repite el tiempo de espera entre los puntos de control antes de continuar.

- Es posible omitir un punto de control bajo ciertas circunstancias. Por ejemplo, supongamos que el grupo de Auto Scaling tiene dos instancias y los porcentajes de puntos de control son `[10,40,100]`. Una vez reemplazada la primera instancia, Amazon EC2 Auto Scaling calcula que se reemplazó el 50 por ciento del grupo. Debido a que el 50 por ciento es mayor que los dos primeros puntos de control, omite el primer punto de control (`10`) y envía una notificación para el segundo punto de control (`40`).
- La cancelación de la operación impide que se realicen nuevos reemplazos. Si cancela la operación o esta genera un error antes de llegar al último punto de control, las instancias que ya se hayan reemplazado no revierten a su configuración anterior.
- En el caso de una actualización parcial, al volver a ejecutar la operación, Amazon EC2 Auto Scaling no se reinicia desde el último punto de comprobación ni se detiene cuando solo se reemplazan las instancias anteriores. Sin embargo, reemplaza primero las instancias anteriores antes de las nuevas instancias.
- El porcentaje real completado puede ser superior al porcentaje de ese punto de control cuando el porcentaje del punto de control es demasiado bajo en relación con la cantidad de instancias del grupo. Por ejemplo, supongamos que el porcentaje del punto de control es del 20 por ciento y que el grupo tiene cuatro instancias. Si Amazon EC2 Auto Scaling reemplaza una de las cuatro instancias, el porcentaje real reemplazado (25 por ciento) será superior al porcentaje del punto de control (20 por ciento).
- Una vez alcanzado un punto de control, el porcentaje total completado que se muestra no se actualiza hasta que las instancias terminan la preparación. Por ejemplo, los porcentajes de punto de control son `[20,50]` con un retraso de punto de control de 15 minutos y un porcentaje de buen estado mínimo del 80 por ciento. El grupo de escalado automático tiene 10 instancias y realiza los siguientes reemplazos:
  - 0:00: dos instancias anteriores se reemplazan por otras nuevas.
  - 0:10: dos instancias nuevas terminan la preparación.
  - 0:25: dos instancias anteriores se reemplazan por otras nuevas. (Solo se reemplazan dos instancias para mantener el porcentaje mínimo en buen estado).
  - 0:35: dos instancias nuevas terminan la preparación.
  - 0:35: una instancia anterior se reemplaza por una nueva.
  - 0:45: una instancia nueva termina la preparación.

A las 0:35, la operación deja de lanzar instancias nuevas. El porcentaje completado aún no refleja con precisión el número de reemplazos completados (50 por ciento), porque la nueva instancia no ha terminado la preparación. Una vez que la nueva instancia completa su período de preparación a las 0:45, el porcentaje completado muestra el 50 por ciento.

- Si hay varios puntos de control habilitados, el punto de control de actualización de la instancia debe estar configurado al 100 % para garantizar la sustitución completa de todas las instancias. Si establece el punto de control final en un porcentaje inferior al 100 % (por ejemplo, el 50 %), la actualización de la instancia dejará de reemplazar las instancias una vez alcanzado el porcentaje de ese punto de control y no continuará automáticamente hasta completar el 100 %.

Habilite los puntos de control mediante el uso de o Consola de administración de AWS AWS CLI

Puede utilizar la Consola de administración de AWS o AWS CLI para activar los puntos de control.

Habilitar puntos de control (consola)

Es posible habilitar los puntos de control antes de iniciar una actualización de instancias para reemplazar instancias mediante un enfoque progresivo o gradual. Esto proporciona tiempo adicional para la verificación.

Para comenzar una actualización de instancias que utiliza puntos de control

1. Abra la consola Amazon EC2 en https://console.aws.amazon.com/ec2/y seleccione Auto Scaling Groups en el panel de navegación.
2. Seleccione la casilla situada junto al grupo de escalado automático.

   Se abre un panel dividido en la parte inferior de la página Auto Scaling groups (Grupos de Auto Scaling).
3. En la pestaña Instance refresh (Actualización de instancias) en Active Instance refresh (Actualización de instancias activas), elija Start instance refresh (Iniciar actualización de instancias).
4. En la página Start instance refresh (Iniciar actualización de instancias), ingrese los valores Minimum healthy percentage (Porcentaje mínimo en buen estado) e Instance warmup (Preparación de la instancia).
5. Seleccione la casilla de verificación Enable checkpoints (Habilitar puntos de control).

Esto muestra un cuadro donde puede definir el porcentaje de umbral para el primer punto de control.

6. En Proceed until ____ % of the group is refreshed (Continuar hasta que se actualice el ____ % del grupo), ingrese un número (1-100). Esto configura el porcentaje del primer punto de control.
7. Para agregar otro punto de control, elija Add checkpoint (Agregar punto de control) y, a continuación, defina el porcentaje para el siguiente punto de control.
8. Para especificar cuánto tiempo espera Amazon EC2 Auto Scaling una vez alcanzado un punto de control, actualice los campos de Wait for **1 hour** between checkpoints (Esperar 1 hora entre puntos de control). La unidad de tiempo puede ser horas, minutos o segundos.
9. Si terminó con las selecciones de actualización de instancias, elija Iniciar actualización de instancias.

Habilitar puntos de control (AWS CLI)

Para iniciar una actualización de instancias con los puntos de control habilitados mediante el AWS CLI, necesita un archivo de configuración que defina los siguientes parámetros:

- `CheckpointPercentages`: especifica los valores de umbral para el porcentaje de instancias que se van a reemplazar. Estos valores de umbral proporcionan los puntos de control. Cuando el porcentaje de instancias que se han reemplazado y preparado alcanza uno de los umbrales especificados, la operación espera un período de tiempo especificado. Para especificar el número de segundos que se debe esperar en `CheckpointDelay`. Una vez transcurrido el período de tiempo especificado, la actualización de instancias continúa hasta que llega al siguiente punto de control (si corresponde).
- `CheckpointDelay`: especifica la cantidad de tiempo, en segundos, que debe esperar después de alcanzar un punto de control antes de continuar. Elija un período de tiempo que proporcione tiempo suficiente para realizar las verificaciones.

El último valor que se muestra en la matriz `CheckpointPercentages` describe el porcentaje del grupo de Auto Scaling que debe reemplazarse correctamente. La operación pasa a `Successful` después de que este porcentaje se reemplaza correctamente y se considera que todas las instancias han terminado de inicializarse.

Para crear varios puntos de control

Para crear varios puntos de control, usa el siguiente comando de ejemplo start-instance-refresh. En este ejemplo se configura una actualización de instancias que actualiza inicialmente el uno por ciento del grupo de Auto Scaling. Después de esperar 10 minutos, se actualiza el 19 por ciento siguiente y espera otros 10 minutos. Por último, actualiza el resto del grupo antes de concluir la operación.

```
aws autoscaling start-instance-refresh --cli-input-json file://config.json
```

Contenido de `config.json`:

```
{
    "AutoScalingGroupName": "my-asg",
    "Preferences": {
      "InstanceWarmup": 60,
      "MinHealthyPercentage": 80,
      "CheckpointPercentages": [1,20,100],
      "CheckpointDelay": 600
    }
}
```

Para crear un único punto de control

Para crear un único punto de control, utilice el siguiente comando de ejemplo start-instance-refresh. En este ejemplo se configura una actualización de instancias que actualiza inicialmente el 20 por ciento del grupo de Auto Scaling. Después de esperar 10 minutos, actualiza el resto del grupo antes de concluir la operación.

```
aws autoscaling start-instance-refresh --cli-input-json file://config.json
```

Contenido de `config.json`:

```
{
    "AutoScalingGroupName": "my-asg",
    "Preferences": {
      "InstanceWarmup": 60,
      "MinHealthyPercentage": 80,
      "CheckpointPercentages": [20,100],
      "CheckpointDelay": 600
    }
}
```

Para actualizar parcialmente el grupo de Auto Scaling

Para reemplazar solo una parte del grupo de Auto Scaling y luego detenerlo por completo, utilice el siguiente start-instance-refreshcomando de ejemplo. En este ejemplo se configura una actualización de instancias que actualiza inicialmente el uno por ciento del grupo de Auto Scaling. Después de esperar 10 minutos, actualiza el siguiente 19 por ciento antes de concluir la operación.

```
aws autoscaling start-instance-refresh --cli-input-json file://config.json
```

Contenido de `config.json`:

```
{
    "AutoScalingGroupName": "my-asg",
    "Preferences": {
      "InstanceWarmup": 60,
      "MinHealthyPercentage": 80,
      "CheckpointPercentages": [1,20],
      "CheckpointDelay": 600
    }
}
```

Reemplazo de instancias de Auto Scaling en función de la duración máxima de la instancia

La duración máxima de la instancia especifica la cantidad máxima de tiempo (en segundos) que una instancia puede estar en servicio antes de que se termine y se sustituya. Un caso de uso común podría ser un requisito para reemplazar las instancias según una programación debido a políticas de seguridad internas o controles de conformidad externos.

Debe especificar un valor de al menos 86 400 segundos (un día). Para borrar un valor establecido anteriormente, especifique un valor nuevo de 0. Esta configuración se aplica a todas las instancias actuales y futuras del grupo de Auto Scaling.

Contenido

- Consideraciones
- Configuración de la duración máxima de la instancia
- Limitaciones

Consideraciones

A continuación, se detallan consideraciones que se deben tener a la hora de utilizar esta característica:

- Cada vez que se reemplaza una instancia anterior y se lanza una instancia nueva, la instancia nueva utiliza la plantilla de lanzamiento o la configuración de lanzamiento asociada actualmente al grupo de escalado automático. Si la plantilla de lanzamiento o la configuración de lanzamiento especifican el ID de imagen de máquina de Amazon (AMI) de una versión diferente de la aplicación, esta versión de la aplicación se implementará de forma automática.
- Si se establece un valor de duración máxima de la instancia demasiado bajo, las instancias pueden ser reemplazadas más rápido de lo deseado. En general, Amazon EC2 Auto Scaling reemplaza las instancias de una en una, con una pausa entre reemplazos. Sin embargo, si la duración máxima de la instancia que especifique no proporciona tiempo suficiente para reemplazar cada instancia de forma individual, Amazon EC2 Auto Scaling debe reemplazar más de una instancia a la vez. Es posible que se reemplacen varias instancias a la vez, hasta un 10 por ciento de la capacidad actual del grupo de Auto Scaling. Para evitar reemplazar demasiadas instancias a la vez, establezca una duración máxima de la instancia más larga o utilice la protección contra la reducción horizontal de instancias para evitar temporalmente que las instancias individuales terminen. Para obtener más información, consulte Uso de la protección frente a la reducción horizontal de instancias para controlar la terminación de instancias.
- De manera predeterminada, Amazon EC2 Auto Scaling crea una nueva actividad de escalado para terminar la instancia y, a continuación, la termina. Mientras se termina la instancia, otra actividad de escalado lanza una instancia nueva. Puede cambiar este comportamiento para que se lance antes de la finalización mediante una política de mantenimiento de instancias. Para obtener más información, consulte Políticas de mantenimiento de instancias.

Configuración de la duración máxima de la instancia

Cuando se crea un grupo de Auto Scaling en la consola, no se puede configurar la duración máxima de la instancia. Sin embargo, una vez que se crea el grupo, se puede editar para configurar la duración máxima de la instancia.

Para configurar la duración máxima de las instancias de un grupo (consola)

1. Abra la consola Amazon EC2 en https://console.aws.amazon.com/ec2/y seleccione Auto Scaling Groups en el panel de navegación.

2. Seleccione la casilla situada junto al grupo de escalado automático.

   Se abre un panel dividido en la parte inferior de la página Auto Scaling groups (Grupos de Auto Scaling), que muestra información sobre el grupo que seleccionó.
3. En la pestaña Details (Detalles) elija (Advanced configurations) Configuraciones avanzadas, Edit (Editar).
4. En Maximum instance lifetime (Duración máxima de la instancia), ingrese la cantidad máxima de segundos que una instancia puede estar en servicio.
5. Elija Actualizar.

En la pestaña Activity (Actividad), en Activity history (Historial de actividad), puede ver el historial de reemplazo de instancias del grupo a lo largo de su historia.

Para configurar la duración máxima de las instancias de un grupo (AWS CLI)

También puede usar AWS CLI para establecer la vida útil máxima de las instancias para grupos de Auto Scaling nuevos o existentes.

Para los nuevos grupos de Auto Scaling, utilice el create-auto-scaling-groupcomando.

```
aws autoscaling create-auto-scaling-group --cli-input-json file://~/config.json
```

A continuación, se incluye un archivo de ejemplo `config.json` que muestra una duración máxima de instancia de `2592000` segundos (30 días).

```
{
    "AutoScalingGroupName": "my-asg",
    "LaunchTemplate": {
        "LaunchTemplateName": "my-launch-template",
        "Version": "$Default"
    },
    "MinSize": 1,
    "MaxSize": 5,
    "MaxInstanceLifetime": 2592000,
    "VPCZoneIdentifier": "subnet-5ea0c127,subnet-6194ea3b,subnet-c934b782",
    "Tags": []
}
```

Para los grupos de Auto Scaling existentes, utilice el update-auto-scaling-groupcomando.

```
aws autoscaling update-auto-scaling-group --auto-scaling-group-name my-existing-asg --
max-instance-lifetime 2592000
```

Para comprobar la duración máxima de las instancias de un grupo de Auto Scaling

Utilice el comando describe-auto-scaling-groups.

```
aws autoscaling describe-auto-scaling-groups --auto-scaling-group-name my-asg
```

Limitaciones

- No se garantiza que la duración máxima sea exacta para cada instancia: no se garantiza que las instancias se sustituyan solo al final de su duración máxima. En algunos casos, es posible que Amazon EC2 Auto Scaling tenga que comenzar a reemplazar las instancias inmediatamente después de actualizar el parámetro de duración máxima de la instancia. El motivo de este comportamiento es evitar que se reemplacen todas las instancias al mismo tiempo.
- Respetar la protección contra la reducción horizontal de instancias: Amazon EC2 Auto Scaling proporciona protección contra la reducción horizontal de instancias para ayudarle a controlar qué instancias puede terminar. Cuando esta protección está habilitada en una instancia, Amazon EC2 Auto Scaling no terminará la instancia aunque haya alcanzado su duración máxima.
- Instancias terminadas antes del lanzamiento: cuando solo hay una instancia en el grupo de escalado automático, la característica de duración máxima de la instancia puede provocar una interrupción, ya que Amazon EC2 Auto Scaling termina una instancia y, a continuación, lanza una instancia nueva de forma predeterminada. Para cambiar este comportamiento a lanzar antes de terminar, consulte Políticas de mantenimiento de instancias.

Aumento o disminución de la capacidad de cómputo de su aplicación con el escalado

Escalado es la posibilidad de aumentar o disminuir la capacidad de cómputo de su aplicación. El escalado comienza con un evento o acción de escalado, que indica a un grupo de escalado automático que lance o termine instancias de Amazon EC2.

Amazon EC2 Auto Scaling ofrece varias formas para ajustar el escalado a fin de que satisfaga mejor las necesidades de sus aplicaciones. Por lo tanto, es importante que conozca bien su aplicación. Tenga en cuenta las siguientes consideraciones:

- ¿Qué papel debe desempeñar Amazon EC2 Auto Scaling en la arquitectura de su aplicación? Es habitual pensar en el escalado automático principalmente como un mecanismo para aumentar y reducir la capacidad; sin embargo, también resulta útil para mantener un número constante de servidores.
- ¿Qué restricciones de costos son importantes para usted? Debido a que Amazon EC2 Auto Scaling utiliza instancias de EC2, solo paga los recursos que utiliza. Conocer las limitaciones de costos le ayuda a decidir cuándo y cuánto escalar sus aplicaciones.
- ¿Qué métricas son importantes para su aplicación? Amazon CloudWatch admite una serie de métricas diferentes que puede usar con su grupo de Auto Scaling.

Contenido

- Elección de su método de escalado
- Establecimiento de límites de escalado para el grupo de escalado automático
- Establecimiento de la preparación predeterminada de instancias para un grupo de escalado automático
- Escalado manual para Amazon EC2 Auto Scaling
- Escalado programado para Amazon EC2 Auto Scaling
- Escalado dinámico para Amazon EC2 Auto Scaling
- Escalado predictivo para Amazon EC2 Auto Scaling
- Control de las instancias de Auto Scaling que se terminan durante una reducción horizontal
- Suspensión y reanudación de procesos de Amazon EC2 Auto Scaling

Elección de su método de escalado

Amazon EC2 Auto Scaling ofrece varias formas de escalar el grupo de escalado automático.

Mantenimiento de un número fijo de instancias

El valor predeterminado para un grupo de escalado automático es no tener ninguna política de escalado adjunta ni acciones programadas, lo que hace que mantenga un tamaño fijo. Una vez que cree su grupo de escalado automático, comienza lanzando un número suficiente de instancias para satisfacer la capacidad deseada. Si no hay condiciones de escalado relacionadas con el grupo, este continúa manteniendo la capacidad deseada en todo momento, aunque una instancia deje de estar en buen estado. Amazon EC2 Auto Scaling también supervisa el estado de cada instancia en su grupo de escalado automático. Cuando encuentra que una instancia está en mal estado, la reemplaza por una nueva instancia. Puede leer una descripción más detallada de este proceso en Comprobaciones de estado para instancias en un grupo de escalado automático.

Escalar manualmente

El escalado manual es la forma más básica de escalar su grupo de escalado automático. Puede actualizar la capacidad deseada del grupo de escalado automático o finalizar las instancias en el grupo de escalado automático. Para obtener más información, consulte Escalado manual para Amazon EC2 Auto Scaling.

Escalado según una programación

El escalado por programación significa que las acciones de escalado se ejecutan de manera automática en función de la fecha y la hora. Esto resulta útil cuando sabe exactamente cuándo aumentar o disminuir el número de instancias del grupo, simplemente porque la necesidad surge de acuerdo con una programación previsible. Para obtener más información, consulte Escalado programado para Amazon EC2 Auto Scaling.

Escalado dinámico en función de la demanda

Una forma más avanzada de escalar los recursos, mediante el escalado dinámico, le permite definir una política de escalado que cambia de forma dinámica el grupo de escalado automático para satisfacer los cambios de demanda. Por ejemplo, supongamos que tiene una aplicación web que actualmente se ejecuta en dos instancias y desea que la utilización de CPU del grupo de escalado automático permanezca en el 50 % cuando cambie la carga en la aplicación. Este método es útil para escalar a medida que se producen cambios en el tráfico, cuando no sabe cuándo cambiará el tráfico. Puede configurar políticas de escalado para que respondan por usted. Existen varios tipos de

políticas (o una combinación de ellas) que puede utilizar para reducir horizontalmente en respuesta a los cambios en el tráfico. Para obtener más información, consulte Escalado dinámico para Amazon EC2 Auto Scaling.

Escalado proactivo

También puede combinar el escalado predictivo y el escalado dinámico (enfoques proactivos y reactivos, respectivamente) para escalar la capacidad de EC2 con mayor rapidez. Utilice el escalado predictivo para aumentar el número de instancias EC2 en su grupo de escalado automático antes de los patrones diarios y semanales de flujos de tráfico. Para obtener más información, consulte Escalado predictivo para Amazon EC2 Auto Scaling.

Establecimiento de límites de escalado para el grupo de escalado automático

Los límites de escalado representan el tamaño de grupo mínimo y máximo que desea para su grupo de escalado automático. Configure los límites por separado para el tamaño mínimo y máximo.

La capacidad deseada del grupo se puede cambiar a un número que esté dentro de los límites de tamaño mínimo y máximo. La capacidad deseada debe ser igual o mayor que el tamaño mínimo del grupo e igual o menor al tamaño máximo del grupo.

- Desired capacity (Capacidad deseada): representa la capacidad inicial del grupo de escalado automático en el momento de su creación. Un grupo de escalado automático intenta mantener la capacidad deseada. Comienza lanzando el número de instancias que se especifican para la capacidad deseada y mantiene este número de instancias siempre que no haya políticas de escalado o acciones programadas asociadas al grupo de escalado automático.
- Minimum capacity (Capacidad mínima): representa el tamaño mínimo del grupo. Cuando se establecen políticas de escalado, estas no pueden reducir la capacidad deseada del grupo por debajo de la capacidad mínima.
- Maximum capacity (Capacidad máxima): representa el tamaño máximo del grupo. Cuando se establecen políticas de escalado, estas no pueden aumentar la capacidad deseada del grupo por encima de la capacidad máxima.

Los límites de tamaño mínimo y máximo también se aplican en las siguientes situaciones:

- Cuando actualiza la capacidad deseada de su grupo de escalado automático manualmente.

- Cuando se ponen en marcha acciones programadas, se actualiza la capacidad deseada. Si se pone en marcha una acción programada sin especificar nuevos límites de tamaño mínimo y máximo para el grupo, se aplicarán los límites de tamaño mínimo y máximo actuales del grupo.

Un grupo de escalado automático intenta mantener siempre la capacidad deseada. En los casos en que una instancia termina inesperadamente (por ejemplo, debido a una interrupción de la instancia de spot, un error de comprobación de estado o una acción humana), el grupo lanza automáticamente una nueva instancia para mantener la capacidad deseada.

Para administrar esta configuración en la consola

1. Abra la consola de Amazon EC2 en https://console.aws.amazon.com/ec2/.
2. En el panel de navegación, seleccione Auto Scaling y elija Auto Scaling Groups (Grupos de Auto Scaling).
3. Desde la página Auto Scaling groups (Grupos de escalado automático), seleccione la casilla de verificación situada junto a su grupo de escalado automático.

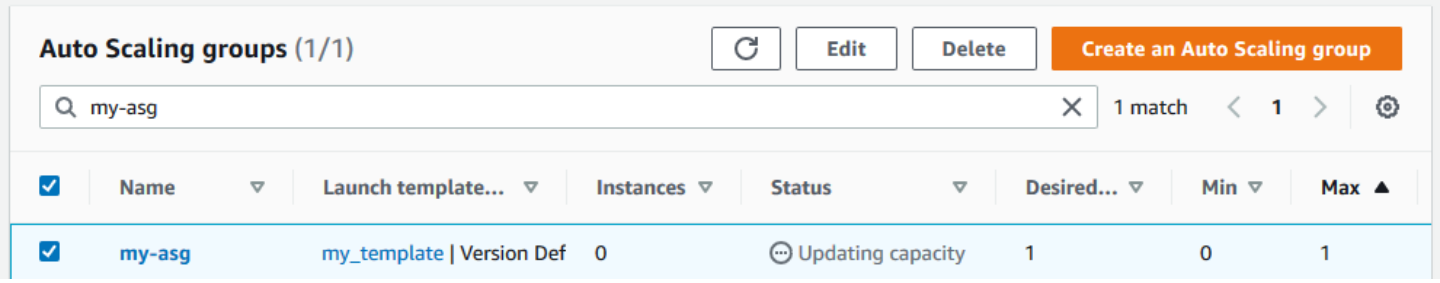
   Se abre un panel dividido en la parte inferior de la página.
4. En el panel inferior, en la pestaña Detalles, consulte o cambie la configuración actual de la capacidad mínima, máxima y deseada del grupo. Para obtener más información, consulte Cambio de la capacidad deseada del grupo de escalado automático.

Por encima del panel de Detalles, puede encontrar información como el número actual de instancias del grupo de escalado automático, la capacidad mínima, máxima y deseada y una columna de estado. Si el grupo de escalado automático utiliza pesos de instancia, la información incluye el número de unidades de capacidad aportadas a la capacidad deseada.

Para agregar o quitar columnas de la lista, elija el icono de configuración en la parte superior de la página. Luego, para los Auto Scaling groups attributes (Atributos de grupos de escalado automático), active o desactive cada columna y elija Confirm (Confirmar).

Para verificar el tamaño del grupo de escalado automático después de realizar cambios

La columna Instances (Instancias) muestra el número de instancias que se están ejecutando actualmente. Cuando una instancia se está lanzando o terminando, la columna Status (Estado) muestra el estado Updating capacity (Actualización de capacidad), como se muestra en la siguiente imagen.



Espere unos minutos y, a continuación, actualice la vista para ver el estado más reciente. Una vez completada una actividad de escalado, la columna Instances (Instancias) muestra un valor nuevo.

Puede ver el número de instancias y el estado de las que se están ejecutando actualmente en la pestaña Instance management (Administración de instancias) de Instances (Instancias).

Establecimiento de la preparación predeterminada de instancias para un grupo de escalado automático

CloudWatch recopila y agrega datos de uso, como E/S de CPU y red, en todas sus instancias de Auto Scaling. Estas métricas se utilizan para crear políticas de escalado que ajustan el número de instancias del grupo de escalado automático a medida que aumenta y disminuye el valor de la métrica seleccionada.

Puede especificar cuánto tiempo esperará después de que una instancia alcance el estado `InService` antes de contribuir con los datos de uso a las métricas agregadas. Este tiempo especificado se denomina preparación predeterminada de instancias. Esto evita que el escalado dinámico se vea afectado por las métricas de instancias individuales que aún no gestionan el tráfico de las aplicaciones y que podrían estar experimentando un alto uso temporal de los recursos de computación.

Para optimizar el rendimiento de las políticas de escalado por pasos y seguimiento de objetivo, le recomendamos encarecidamente que habilite y configure la preparación predeterminada de instancias. No está lista ni configurada de forma predeterminada.

Cuando habilite la preparación predeterminada de instancias, tenga en cuenta que, si su grupo de escalado automático está configurado para usar una política de mantenimiento de instancias, o si usa una actualización de instancias para reemplazar las instancias, también puede evitar que las instancias se cuenten para el porcentaje de buen estado mínimo antes de que hayan terminado de inicializarse.

Contenido

- Consideraciones sobre el rendimiento de escalado
- Elección del tiempo de preparación predeterminada de instancias
- Habilitación de la preparación predeterminada de instancias para un grupo
- Verificación del tiempo de preparación predeterminada de instancias para un grupo
- Búsqueda de políticas de escalado con un tiempo de preparación de instancias previamente establecido
- Borrado de la preparación de instancias previamente establecida para una política de escalado

Consideraciones sobre el rendimiento de escalado

Para la mayoría de las aplicaciones, es útil tener un tiempo de preparación predeterminado de instancias que se aplique a todas las características, en lugar de diferentes tiempos de preparación para las distintas características. Por ejemplo, si no establece una preparación de instancias predeterminada, la característica de actualización de instancias utiliza el período de gracia de la comprobación de estado como tiempo de preparación predeterminado. Si tiene alguna política de seguimiento de objetivos y escalado por pasos, se utilizará el valor establecido para el tiempo de recuperación predeterminado como tiempo de preparación predeterminado. Si tiene alguna política de escalado predictivo, la política no tiene un tiempo de preparación predeterminado.

Si bien las instancias se encuentran en preparación, sus políticas dinámicas de escalado solo escalan horizontalmente si el valor de la métrica de las instancias que no están en preparación es mayor que el umbral superior de la alarma de la política (o la utilización de objetivo de una política de escalado de seguimiento de objetivo). Si la demanda disminuye, el escalado dinámico se vuelve más conservador para proteger la disponibilidad de la aplicación. Esto bloquea las actividades de reducción horizontal para el escalado dinámico hasta que termine la preparación de las instancias nuevas.

Durante el escalado horizontal, Amazon EC2 Auto Scaling considera que las instancias que se están preparando son parte de la capacidad del grupo a la hora de decidir cuántas instancias se van añadiendo al grupo. Por lo tanto, varias infracciones de alarma que requieren añadir una cantidad similar de capacidad producen una única actividad de escalado. La intención es realizar continuamente (pero no excesivamente) un escalado horizontal.

Si el calentamiento de instancias predeterminado no está activado, el tiempo que espera una instancia antes de enviar las métricas CloudWatch y contabilizarlas para su capacidad actual variará

de una instancia a otra. Entonces, existe la posibilidad de que sus políticas de escalado funcionen de manera impredecible en comparación con la carga de trabajo real que se está produciendo.

Por ejemplo, pensemos en una aplicación con un patrón de carga de trabajo recurrente on-and-off. Se utiliza una política de escalado predictivo para tomar decisiones recurrentes sobre si se debe aumentar el número de instancias. Como no hay un tiempo de preparación predeterminado para las políticas de escalado predictivo, las instancias comienzan a contribuir a las métricas agregadas de forma inmediata. Si estas instancias utilizan más recursos al iniciarse, la adición de instancias podría provocar un pico en las métricas globales. En función del tiempo que tarde en estabilizarse el uso, esto podría afectar a cualquier política de escalado dinámico que utilice estas métricas. Si se supera el umbral máximo de alarma establecido en una política de escalado dinámico, el grupo volverá a aumentar de tamaño. Mientras las nuevas instancias se estén preparando, las actividades de reducción horizontal se bloquearán.

Elección del tiempo de preparación predeterminada de instancias

La clave para configurar la preparación predeterminado de las instancias es determinar cuánto tiempo necesitan las instancias para terminar de inicializarse y para que el consumo de recursos se estabilice una vez que alcancen el estado `InService`. Cuando elija un tiempo de la preparación de instancias, busque mantener un equilibrio óptimo entre recopilar datos de uso para tráfico legítimo, a la vez que minimiza la recopilación de datos asociada a los picos de uso temporales durante el inicio.

Supongamos que tiene un grupo de escalado automático conectado a un equilibrador de carga Elastic Load Balancing. Cuando las instancias nuevas terminan de lanzarse, se registran en el equilibrador de carga antes de ingresar al estado de `InService`. Después de las instancias ingresan al estado `InService`, el consumo de recursos puede seguir experimentando picos temporales y necesitar tiempo para estabilizarse. Por ejemplo, el consumo de recursos de un servidor de aplicaciones que debe descargar y almacenar en caché activos de gran tamaño tarda más tiempo en estabilizarse que un servidor web ligero sin grandes recursos para descargar. La preparación de instancias proporciona el retraso de tiempo necesario para que se estabilice el consumo de recursos.

> **Important**
>
> Si no está seguro del tiempo que necesita para el tiempo de preparación, puede empezar con 300 segundos. A continuación, disminuya o aumente gradualmente el tiempo hasta obtener el mejor rendimiento de escalado para su aplicación. Es probable que deba hacer esto varias veces para lograr los resultados deseados. Como alternativa,

si tiene alguna política de escalado que tenga su propio tiempo de preparación (`EstimatedInstanceWarmup`), puede usar este valor para empezar. Para obtener más información, consulte Búsqueda de políticas de escalado con un tiempo de preparación de instancias previamente establecido.

Considere utilizar enlaces de ciclo de vida para casos de uso en los que tenga scripts o tareas de configuración para que se ejecuten en el inicio. Los enlaces de ciclo de vida pueden retrasar la puesta en servicio de las instancias nuevas hasta que hayan terminado de inicializarse. Resultan especialmente útiles si tiene scripts de arranque que tardan un poco en completarse. Si agrega un enlace de ciclo de vida, puede reducir el valor de la preparación predeterminada de instancias. Para obtener más información acerca del uso de enlaces de ciclo de vida, consulte Enlaces de ciclo de vida de Amazon EC2 Auto Scaling.

Habilitación de la preparación predeterminada de instancias para un grupo

Puede habilitar la preparación predeterminada de instancias cuando cree un grupo de escalado automático. También puede habilitarla para grupos existentes.

Al habilitar la característica de preparación predeterminada de instancias, ya no tiene que especificar valores para los parámetros de preparación de las siguientes características:

- Actualización de instancias
- Escalado de seguimiento de destino
- Escalado por pasos

Console

Para habilitar la preparación predeterminada de instancias para un nuevo grupo (consola)

Al crear el grupo de escalado automático, en la página Configure advanced options (Configurar las opciones avanzadas), en Additional settings (Configuración adicional), seleccione la opción Enable default instance warmup (Habilitar preparación predeterminada de instancias). Elija el tiempo de preparación que necesite para la aplicación.

AWS CLI

Para habilitar la preparación predeterminada de instancias para un nuevo grupo (AWS CLI)

Para habilitar la preparación predeterminada de instancias para un grupo de escalado automático, agregue la opción `--default-instance-warmup` y especifique un valor, en segundos, de 0 a 3600. Una vez habilitada, un valor de `-1` desactivará esta configuración.

El siguiente create-auto-scaling-groupcomando crea un grupo de Auto Scaling con el nombre *my-asg* y habilita el calentamiento de instancias predeterminado con un valor de *120* segundos.

```
aws autoscaling create-auto-scaling-group --auto-scaling-group-name my-asg --
default-instance-warmup 120 ...
```

> **Tip**
>
> Si este comando arroja un error, asegúrate de haber actualizado la versión AWS CLI local a la última versión.

Console

Para habilitar la preparación predeterminada de instancias para un grupo existente (consola)

1. Abra la consola Amazon EC2 en https://console.aws.amazon.com/ec2/y seleccione Auto Scaling Groups en el panel de navegación.
2. En la barra de navegación de la parte superior de la pantalla, elija la misma Región de AWS que utilizó cuando creó el grupo de escalado automático.
3. Seleccione la casilla situada junto al grupo de escalado automático.

   Se abre un panel dividido en la parte inferior de la página.
4. En la pestaña Details (Detalles) elija (Advanced configurations) Configuraciones avanzadas, Edit (Editar).
5. Para Preparación predeterminada de instancias, elija el tiempo de preparación que necesite para la aplicación.
6. Elija Actualizar.

AWS CLI

Para habilitar la preparación predeterminada de instancias para un grupo existente (AWS CLI)

El siguiente ejemplo usa el update-auto-scaling-groupcomando para habilitar el calentamiento de instancias predeterminado con un valor de *120* segundos para un grupo de Auto Scaling existente denominado. *my-asg*

```
aws autoscaling update-auto-scaling-group --auto-scaling-group-name my-asg --
default-instance-warmup 120
```

> **Tip**
>
> Si este comando arroja un error, asegúrate de haber actualizado la versión AWS CLI local a la última versión.

Verificación del tiempo de preparación predeterminada de instancias para un grupo

Utilice el siguiente procedimiento para verificar el tiempo de preparación predeterminada de instancias para un grupo de escalado automático con la AWS CLI.

Cómo verificar el tiempo de preparación predeterminada de instancias para un grupo de escalado automático

Use el siguiente comando describe-auto-scaling-groups. *my-asg*Sustitúyalo por el nombre de tu grupo de Auto Scaling.

```
aws autoscaling describe-auto-scaling-groups --auto-scaling-group-name my-asg
```

A continuación, se muestra un ejemplo de respuesta.

```
{
    "AutoScalingGroups": [
        {
            "AutoScalingGroupName": "my-asg",
            "AutoScalingGroupARN": "arn",
            ...
            "DefaultInstanceWarmup": 120
        }
    ]
}
```

Búsqueda de políticas de escalado con un tiempo de preparación de instancias previamente establecido

Para identificar si tiene políticas que tienen su propio tiempo de preparación de `EstimatedInstanceWarmup`, ejecute el siguiente comando describe-policies con la AWS CLI. *my-asg*Sustitúyalo por el nombre de tu grupo de Auto Scaling.

```
aws autoscaling describe-policies --auto-scaling-group-name my-asg
  --query 'ScalingPolicies[?EstimatedInstanceWarmup!=`null`]'
```

A continuación, se muestra un ejemplo del resultado.

```
[
  {
    "AutoScalingGroupName":"my-asg",
    "PolicyName":"cpu50-target-tracking-scaling-policy",
    "PolicyARN":"arn",
    "PolicyType":"TargetTrackingScaling",
    "StepAdjustments":[],
    "EstimatedInstanceWarmup":120,
    "Alarms":[{
        "AlarmARN": "arn:aws:cloudwatch:us-west-2:123456789012:alarm:TargetTracking-my-
asg-AlarmHigh-fc0e4183-23ac-497e-9992-691c9980c38e",
        "AlarmName": "TargetTracking-my-asg-AlarmHigh-
fc0e4183-23ac-497e-9992-691c9980c38e"
      },
      {
            "AlarmARN": "arn:aws:cloudwatch:us-
west-2:123456789012:alarm:TargetTracking-my-asg-AlarmLow-61a39305-ed0c-47af-
bd9e-471a352ee1a2",
            "AlarmName": "TargetTracking-my-asg-AlarmLow-61a39305-ed0c-47af-
bd9e-471a352ee1a2"
    }],
    "TargetTrackingConfiguration":{
      "PredefinedMetricSpecification":{
        "PredefinedMetricType":"ASGAverageCPUUtilization"
      },
      "TargetValue":50.0,
      "DisableScaleIn":false
    },
    "Enabled":true
  },
```

```
    ... additional policies ...

]
```

Borrado de la preparación de instancias previamente establecida para una política de escalado

Tras habilitar la preparación de instancias predeterminada, actualice las políticas de escalado que aún tengan su propio tiempo de preparación para borrar el valor establecido anteriormente. De lo contrario, anulará la preparación de instancias predeterminada.

Puede actualizar las políticas de escalado mediante la consola AWS CLI, o AWS SDKs. En esta sección se describen los pasos de la consola. Si usa AWS CLI o AWS SDKs, asegúrese de conservar la configuración de políticas existente, pero elimine la `EstimatedInstanceWarmup` propiedad. Al actualizar una política de escalado existente, la política se sustituirá por la que especifique al realizar la llamada mediante programación. PutScalingPolicy Los valores originales no se conservan.

Borrado de la preparación de instancias establecida anteriormente para una política de escalado (consola)

1. Abra la consola Amazon EC2 en https://console.aws.amazon.com/ec2/y seleccione Auto Scaling Groups en el panel de navegación.
2. Seleccione la casilla situada junto al grupo de escalado automático.

   Se abre un panel dividido en la parte inferior de la página.
3. En la pestaña Escalado automático, en Políticas de escalado dinámico, elija la política que le interese y, a continuación, seleccione Acciones y Editar.
4. En Preparación de instancias, desactive el valor de preparación de instancias para usar en su lugar el valor predeterminado de preparación de instancias.
5. Elija Actualizar.

Escalado manual para Amazon EC2 Auto Scaling

Puede ajustar manualmente el número de instancias de EC2 en su grupo de escalado automático en cualquier momento. Este proceso de cambiar el recuento de instancias manualmente se denomina

escalado manual. El escalado manual es una alternativa al escalado automático, especialmente si desea realizar cambios de capacidad únicos.

Tras escalar el grupo manualmente, Amazon EC2 Auto Scaling reanuda las actividades normales de escalado automático en función de las políticas de escalado y las acciones programadas que haya definido. En el caso de los grupos en las que la preparación predeterminada de instancias esté activada, las instancias nuevas pasan por un período de preparación antes de empezar a contribuir a las métricas utilizadas para el escalado automático. Este período de preparación ayuda a estabilizar el grupo en la nueva capacidad. Para obtener más información, consulte Establecimiento de la preparación predeterminada de instancias para un grupo de escalado automático.

A veces, es posible que quiera deshabilitar temporalmente las políticas de escalado y las acciones programadas antes de escalar manualmente un grupo. De este modo, evita que surjan conflictos entre las acciones de escalado manual y las actividades de escalado automatizado. Para obtener más información, consulte Desactivación de las actividades de escalado.

Contenido

- Cambio de la capacidad deseada del grupo de escalado automático
- Finalización de una instancia en su grupo de escalado automático (AWS CLI)

Cambio de la capacidad deseada del grupo de escalado automático

Cuando cambia la capacidad del grupo de escalado automático, Amazon EC2 Auto Scaling administra el proceso de lanzar y terminar instancias para alcanzar el nuevo tamaño deseado.

Console

Para cambiar el tamaño del grupo de escalado automático

1. Abra la consola Amazon EC2 en https://console.aws.amazon.com/ec2/y seleccione Auto Scaling Groups en el panel de navegación.
2. Seleccione la casilla situada junto al grupo de escalado automático.

   Se muestra un panel dividido en la parte inferior de la página.
3. En la pestañaDetails (Detalles) elija Group details (Detalles de grupo), Edit (Editar).
4. En Capacidad deseada, aumente o disminuya la capacidad deseada. Por ejemplo, para aumentar el tamaño del grupo en uno, si el valor actual es 1, introduzca 2.

Si su nuevo valor para la Capacidad deseada, es superior a la Capacidad deseada mínima y a la Capacidad deseada máxima, la Capacidad deseada máxima se incrementará automáticamente al nuevo valor de capacidad deseada.

5. Elija Update (Actualizar) cuando haya terminado.

Compruebe que el tamaño del grupo que especificó haya provocado el lanzamiento de la misma cantidad de instancias. Por ejemplo, si ha aumentado el tamaño del grupo en uno, verifique que el grupo de escalado automático haya lanzado una instancia adicional.

Para verificar que el tamaño del grupo de escalado automático ha cambiado

1. En la pestaña Actividad, en Historial de actividad, puede ver el progreso de las actividades que están asociadas al grupo de escalado automático. La columna Status (Estado) muestra el estado actual de su instancia. Mientras se está lanzando la instancia, la columna de estado muestra `Not yet in service`. El estado cambia a `Successful` cuando se lanza la instancia. También puede utilizar el icono de actualización para ver el estado actual de la instancia. Para obtener más información, consulte Verificación de una actividad de escalado para un grupo de escalado automático.
2. En la pestaña Administración de instancia, en Instancias, puede ver el estado de la instancia. La instancia tarda poco tiempo en lanzarse.
   - La columna Lifecycle (Ciclo de vida) muestra el estado de su instancia. Al principio, la instancia tiene el estado `Pending`. Cuando una instancia está lista para recibir tráfico, su estado es `InService`.
   - La columna Estado muestra el resultado de la comprobación de estado de Amazon EC2 Auto Scaling correspondiente a su instancia.

AWS CLI

En el siguiente ejemplo se presupone que ha creado un grupo de escalado automático con un tamaño mínimo de 1 y un tamaño máximo de 5. Por lo tanto, el grupo tiene una sola instancia en ejecución.

Para cambiar el tamaño del grupo de escalado automático

Use el set-desired-capacitycomando para cambiar el tamaño del grupo de Auto Scaling, como se muestra en el siguiente ejemplo.

```
aws autoscaling set-desired-capacity --auto-scaling-group-name my-asg \
  --desired-capacity 2
```

Si opta por respetar el periodo de recuperación predeterminado del grupo de escalado automático, debe especificar la opción `--honor-cooldown` tal y como se muestra en el ejemplo siguiente. Para obtener más información, consulte Recuperaciones de escalado para Amazon EC2 Auto Scaling.

```
aws autoscaling set-desired-capacity --auto-scaling-group-name my-asg \
  --desired-capacity 2 --honor-cooldown
```

Para verificar el tamaño del grupo de escalado automático

Use el describe-auto-scaling-groupscomando para confirmar que el tamaño del grupo de Auto Scaling ha cambiado, como en el siguiente ejemplo.

```
aws autoscaling describe-auto-scaling-groups --auto-scaling-group-name my-asg
```

A continuación, se muestra un resultado de ejemplo que muestra detalles sobre el grupo y las instancias lanzadas.

```
{
    "AutoScalingGroups": [
        {
            "AutoScalingGroupName": "my-asg",
            "AutoScalingGroupARN": "arn",
            "LaunchTemplate": {
                "LaunchTemplateName": "my-launch-template",
                "Version": "1",
                "LaunchTemplateId": "lt-050555ad16a3f9c7f"
            },
            "MinSize": 1,
            "MaxSize": 5,
            "DesiredCapacity": 2,
            "DefaultCooldown": 300,
            "AvailabilityZones": [
                "us-west-2a"
            ],
            "LoadBalancerNames": [],
            "TargetGroupARNs": [],
            "HealthCheckType": "EC2",
```

```
            "HealthCheckGracePeriod": 300,
            "Instances": [
                {
                    "ProtectedFromScaleIn": false,
                    "AvailabilityZone": "us-west-2a",
                    "LaunchTemplate": {
                        "LaunchTemplateName": "my-launch-template",
                        "Version": "1",
                        "LaunchTemplateId": "lt-050555ad16a3f9c7f"
                    },
                    "InstanceId": "i-05b4f7d5be44822a6",
                    "InstanceType": "t3.micro",
                    "HealthStatus": "Healthy",
                    "LifecycleState": "Pending"
                },
                {
                    "ProtectedFromScaleIn": false,
                    "AvailabilityZone": "us-west-2a",
                    "LaunchTemplate": {
                        "LaunchTemplateName": "my-launch-template",
                        "Version": "1",
                        "LaunchTemplateId": "lt-050555ad16a3f9c7f"
                    },
                    "InstanceId": "i-0c20ac468fa3049e8",
                    "InstanceType": "t3.micro",
                    "HealthStatus": "Healthy",
                    "LifecycleState": "InService"
                }
            ],
            "CreatedTime": "2019-03-18T23:30:42.611Z",
            "SuspendedProcesses": [],
            "VPCZoneIdentifier": "subnet-c87f2be0",
            "EnabledMetrics": [],
            "Tags": [],
            "TerminationPolicies": [
                "Default"
            ],
            "NewInstancesProtectedFromScaleIn": false,
            "ServiceLinkedRoleARN": "arn",
            "TrafficSources": []
        }
    ]
}
```

Observe que `DesiredCapacity` muestra el nuevo valor. El grupo de escalado automático ha lanzado una instancia adicional.

Finalización de una instancia en su grupo de escalado automático (AWS CLI)

Hay ocasiones en las que es posible que desee reducir horizontalmente de modo manual su grupo de escalado automático, pero desee terminar una instancia específica. Puede escalar manualmente su grupo de Auto Scaling mediante el comando terminate-instance-in-auto-scaling-group y especificando el ID de la instancia que desea terminar y la `--should-decrement-desired-capacity` opción, como se muestra en el siguiente ejemplo.

```
aws autoscaling terminate-instance-in-auto-scaling-group \
  --instance-id i-026e4c9f62c3e448c --should-decrement-desired-capacity
```

A continuación, se muestra un resultado de ejemplo que muestra detalles sobre la actividad de escalado.

```
{
    "Activities": [
        {
            "ActivityId": "b8d62b03-10d8-9df4-7377-e464ab6bd0cb",
            "AutoScalingGroupName": "my-asg",
            "Description": "Terminating EC2 instance: i-026e4c9f62c3e448c",
            "Cause": "At 2023-09-23T06:39:59Z instance i-026e4c9f62c3e448c was taken
 out of service in response to a user request, shrinking the capacity from 1 to 0.",
            "StartTime": "2023-09-23T06:39:59.015000+00:00",
            "StatusCode": "InProgress",
            "Progress": 0,
            "Details": "{\"Subnet ID\":\"subnet-6194ea3b\",\"Availability Zone\":\"us-
west-2c\"}"
        }
    ]
}
```

Esta opción no está disponible en la consola. Sin embargo, puede utilizar la página Instancias de la consola de Amazon EC2 para finalizar una instancia de su grupo de escalado automático. Al hacerlo, Amazon EC2 Auto Scaling detecta que la instancia ya no se ejecuta y la reemplaza automáticamente como parte del proceso de comprobación de estado. Tras finalizar la instancia, transcurren uno o

dos minutos antes de que se lance una nueva instancia. Para obtener más información sobre la terminación de una instancia, consulte Terminar una instancia en la Guía del usuario de Amazon EC2.

Si termina instancias en su grupo y eso provoca una distribución desigual entre las zonas de disponibilidad, Amazon EC2 Auto Scaling reequilibra el grupo para restablecer una distribución uniforme, a menos que suspenda el proceso de `AZRebalance`. Para obtener más información, consulte Suspensión y reanudación de procesos de Amazon EC2 Auto Scaling.

Escalado programado para Amazon EC2 Auto Scaling

Con el escalado automático programado, puede configurar escalados automáticos para la aplicación según cambios predecibles en las cargas. Puede crear acciones programadas que aumentan o disminuyen la capacidad deseada del grupo en momentos específicos.

Por ejemplo, supongamos que experimenta un patrón de tráfico semanal regular en el que la carga aumenta a mitad de semana y disminuye hacia el final de la semana. Puede configurar un programa de escalado en Amazon EC2 Auto Scaling que se alinee con este patrón:

- El miércoles por la mañana, una acción programada aumentará la capacidad al aumentar la capacidad deseada previamente establecida del grupo de escalado automático.
- El viernes por la noche, otra acción programada reducirá la capacidad al disminuir la capacidad deseada previamente establecida del grupo de escalado automático.

Estas acciones de escalado programadas le permiten optimizar los costes y el rendimiento. Su aplicación tiene la capacidad suficiente para gestionar los picos de tráfico a mitad de semana, pero no aprovisiona en exceso la capacidad innecesaria en otros momentos.

Puede combinar el escalado programado y las políticas de escalado para obtener los beneficios de ambos enfoques de escalado. Después de ejecutar una acción de escalado programado, la política de escalado puede seguir tomando decisiones sobre si desea ampliar la capacidad. Esto le ayuda a garantizar que tiene capacidad suficiente para controlar la carga de su aplicación. Mientras la aplicación se escala para adaptarse a la demanda, la capacidad actual debe estar dentro de la capacidad mínima y máxima establecida por la acción programada.

Contenido

- Cómo funciona el escalado programado

- Programas recurrentes
- Zona horaria
- Consideraciones
- Limitaciones
- Creación de una acción programada
- Cómo ver detalles de acciones programadas
- Eliminación de una acción programada

Cómo funciona el escalado programado

Para utilizar el escalado programado, cree acciones programadas que indican a Amazon EC2 Auto Scaling que realice actividades de escalado en momentos específicos. Cuando crea una acción programada, especifica el grupo de escalado automático, cuándo se debe producir la actividad de escalado, la nueva capacidad deseada y, opcionalmente, las nuevas capacidades mínima y máxima. Puede crear acciones programadas que se escalen solo una vez o que se escalen de forma periódica.

Cuando llega la hora especificada, Amazon EC2 Auto Scaling escala según los nuevos valores de capacidad, comparando la capacidad actual con la capacidad deseada especificada.

- Si la capacidad actual es inferior a la capacidad especificada, Amazon EC2 Auto Scaling escala horizontalmente o añade instancias hasta alcanzar la capacidad deseada especificada.
- Si la capacidad actual es superior a la capacidad especificada, Amazon EC2 Auto Scaling reduce horizontalmente o elimina instancias hasta alcanzar la capacidad deseada especificada.

Una acción programada establece los tamaños mínimo, máximo y deseado del grupo en la fecha y hora especificadas. Puede crear una acción programada solo para una de estas capacidades a la vez, por ejemplo, la capacidad deseada. Sin embargo, en algunos casos, debe incluir las capacidades mínima y máxima para asegurarse de que la capacidad deseada que especificó en la acción no está fuera de estos límites.

Programas recurrentes

Para crear una programación recurrente mediante el SDK AWS CLI o el SDK, especifica una expresión cron y una zona horaria para describir cuándo se repetirá la acción programada.

Opcionalmente, puede especificar una fecha y una hora para la hora de inicio, la hora de finalización o para ambas.

Para crear una programación periódica mediante Consola de administración de AWS, especifique el patrón de recurrencia, la zona horaria, la hora de inicio y la hora de finalización opcional de la acción programada. Todas las opciones de patrón de recurrencia se basan en expresiones cron. Alternativamente, puede escribir su propia expresión cron personalizada.

El formato de expresión cron admitido consta de cinco campos separados por espacios en blanco: [Minuto] [Hora] [Día_del_mes] [Mes_del_año] [Día_de_la_semana]. Por ejemplo, la expresión cron `30 6 * * 2` configura una acción programada que se repite cada martes a las 06:30. El asterisco se utiliza como comodín para coincidir con todos los valores de un campo. Para ver otros ejemplos de expresiones cron, consulte https://crontab.guru/examples.html. Para obtener información sobre cómo escribir sus propias expresiones cron en este formato, consulte Crontab.

Elija cuidadosamente sus horarios de inicio y fin. Tenga en cuenta lo siguiente:

- Si especifica una hora de inicio, Amazon EC2 Auto Scaling realiza la acción en ese momento y, a continuación, realiza la acción basada en la recurrencia especificada.
- Si especifica una hora de finalización, la acción deja de repetirse después de esta hora. Una acción programada no se mantiene en su cuenta una vez que ha alcanzado su hora de finalización.
- Si la hora de repetición coincide exactamente con la hora de finalización, Amazon EC2 Auto Scaling no realizará la acción programada a la hora de finalización.
- La hora de inicio y la hora de finalización deben estar configuradas en UTC cuando utilices el SDK AWS CLI o un SDK.

Zona horaria

De forma predeterminada, las programaciones recurrentes se establecen en Hora universal coordinada (UTC). Puede cambiar la zona hora para que se corresponda con la zona horaria local o con una zona horaria de otra parte de la red. Cuando se especifica una zona horaria que observa el horario de verano (DST), la acción se ajusta automáticamente para horario de verano.

Los valores válidos son los nombres canónicos de las zonas horarias de la base de datos de zonas horarias de la Internet Assigned Numbers Authority (IANA). Por ejemplo, la hora del este de EE. UU. se identifica canónicamente como `America/New_York`. Para obtener más información, consulte https://www.iana.org/time-zones.

Las zonas horarias basadas en la ubicación, como `America/New_York`, se ajustan automáticamente al DST (horario de verano). Sin embargo, una zona horaria basada en UTC como `Etc/UTC` es una hora absoluta y no se ajustará al horario de verano.

Por ejemplo, tiene una programación recurrente cuya zona horaria es `America/New_York`. La primera acción de escalado tiene lugar en la zona horaria `America/New_York` antes de que comience el horario de verano. La siguiente acción de escalado ocurre en la zona horaria `America/New_York` después de que se inicie el horario de verano. La primera acción comienza a las 8:00 UTC-5 en hora local, mientras que la segunda comienza a las 8:00 UTC-4 en hora local.

Si creas una acción programada con el horario de verano Consola de administración de AWS y especificas una zona horaria que respete el horario de verano, tanto la programación periódica como las horas de inicio y finalización se ajustarán automáticamente al horario de verano.

Consideraciones

Cuando cree una acción programada, tenga en cuenta lo siguiente:

- El orden de ejecución de las acciones programadas se garantiza dentro del mismo grupo, pero no para las acciones programadas en los distintos grupos.
- Una acción programada, por lo general, se ejecuta en cuestión de segundos. Sin embargo, la acción puede retrasarse durante un máximo de dos minutos desde la hora de inicio programada. Como las acciones programadas de un grupo de escalado automático se ejecutan en el orden en el que se especifican, las acciones con horas de inicio programadas cercanas pueden tardar más en ejecutarse.
- Puede desactivar temporalmente el escalado programado para un grupo de escalado automático suspendiendo el proceso de `ScheduledActions`. Esto evita que las acciones programadas estén activas sin tener que eliminarlas. Podrá reanudar el escalado programado cuando desee volver a utilizarlo. Para obtener más información, consulte Suspensión y reanudación de procesos de Amazon EC2 Auto Scaling.
- Después de crear una acción programada, puede actualizar cualquiera de sus configuraciones excepto el nombre.
- Cuando varias acciones programadas dentro del mismo grupo de escalado automático tienen expresiones cron idénticas, el orden de ejecución se vuelve arbitrario e indefinido. A fin de garantizar un comportamiento predecible, puede utilizar las horas únicas de inicio programadas para cada acción programada.

Limitaciones

- Los nombres de las acciones programadas deben ser únicos por grupo de escalado automático.
- Una acción programada debe tener un valor temporal único. Si trata de programar una actividad para una hora en la que ya se ha programado otra actividad de escalado, se rechaza la llamada y se devuelve un error que indica que ya existe una acción programada con esta hora de inicio programada.
- Puede crear un máximo de 125 acciones programadas por grupo de escalado automático.

Creación de una acción programada

Para crear una acción programada para el grupo de escalado automático, use uno de los siguientes métodos:

Console

Para crear una acción programada

1. Abra la consola Amazon EC2 en https://console.aws.amazon.com/ec2/y seleccione Auto Scaling Groups en el panel de navegación.
2. Seleccione la casilla situada junto al grupo de escalado automático.

   Se abre un panel dividido en la parte inferior de la página.
3. En la pestaña Escalado automático, en Acciones programadas, elija Crear acción programada.
4. Escriba un Name (Nombre), para la acción programada.
5. Para Capacidad deseada, Mín., Máx., elija la nueva capacidad deseada del grupo y los nuevos límites de tamaños mínimos y máximos. La capacidad deseada debe ser igual o mayor que el tamaño mínimo del grupo e igual o menor al tamaño máximo del grupo.
6. En Recurrence (Recurrencia), elija una de las opciones disponibles.

   - Si desea escalar según una programación recurrente, elija la frecuencia con la que Amazon EC2 Auto Scaling debe ejecutar la acción programada.
     - Si elige una opción que comienza por Every (Cada), la expresión de cron se crea automáticamente.

- Si elige Cron, escriba una expresión cron que especifique cuándo se debe realizar la acción.
- Si desea escalar una sola vez, elija Once (Una vez).

7. Para Time zone (Zona horaria), elija una zona horaria. El valor predeterminado es `Etc/UTC`.

   Todas las zonas horarias enumeradas provienen de la base de datos de zona horaria de IANA. Para obtener más información, consulte https://en.wikipedia. org/wiki/List_of_tz_database_time_zones.

8. Definir una fecha y hora para Specific start time (Hora de inicio específica).

   - Si elige una programación recurrente, la hora de inicio define cuándo se ejecuta la primera acción programada de la serie recurrente.
   - Si eligió Once (Una vez) para la recurrencia, la hora de inicio define la fecha y la hora para que se ejecute la acción de la programación.

9. (Opcional) Para programaciones recurrentes, puede especificar una hora de finalización seleccionando Set End Time (Configurar hora de finalización) y, a continuación, elegir una fecha y hora para End by (Finalizar el).

10. Seleccione Crear. La consola muestra las acciones programadas para el grupo de escalado automático.

AWS CLI

Para crear una acción programada, puede utilizar uno de los siguientes comandos de ejemplo. Reemplace cada *user input placeholder* por su propia información.

Ejemplo: Escalar solo una vez

Utilice el siguiente comando -action con las opciones y. put-scheduled-update-group `--start-time "YYYY-MM-DDThh:mm:ssZ" --desired-capacity`

```
aws autoscaling put-scheduled-update-group-action --scheduled-action-name my-one-time-action \
  --auto-scaling-group-name my-asg --start-time "2021-03-31T08:00:00Z" --desired-capacity 3
```

Ejemplo: Para programar el escalado de forma periódica

Utilice el siguiente comando put-scheduled-update-group-action con las opciones `--recurrence "cron expression"` y `--desired-capacity`.

```
aws autoscaling put-scheduled-update-group-action --scheduled-action-name my-recurring-action \
  --auto-scaling-group-name my-asg --recurrence "0 9 * * *" --desired-capacity 3
```

De forma predeterminada, Amazon EC2 Auto Scaling ejecuta el programa recurrente especificado en función de la zona horaria UTC. Para especificar una zona horaria diferente, incluya la opción `--time-zone` y el nombre para la zona horaria de IANA, como en el siguiente ejemplo.

```
--time-zone "America/New_York"
```

Para obtener más información, consulte https://en.wikipedia. org/wiki/List_of_tz_database_time_zones.

Cómo ver detalles de acciones programadas

Para ver los detalles de las próximas acciones programadas para el grupo de escalado automático, use uno de los siguientes métodos:

Console

Cómo ver detalles de acciones programadas

1. Abra la consola Amazon EC2 en https://console.aws.amazon.com/ec2/y seleccione Auto Scaling Groups en el panel de navegación.
2. Seleccione el grupo de escalado automático.
3. En la pestaña Escalado automático, en la sección Acciones programadas, puede ver las próximas acciones programadas.

Tenga en cuenta que la consola muestra los valores de Hora de inicio y Hora de finalización en su hora local con el desfase UTC en vigor en la fecha y hora especificadas. El desfase UTC es la diferencia, en horas y minutos, entre la hora local y UTC. El valor de Time zone (Zona horaria) muestra la zona horaria solicitada, por ejemplo, `America/New_York`.

AWS CLI

Use el siguiente comando describe-scheduled-actions.

```
aws autoscaling describe-scheduled-actions --auto-scaling-group-name my-asg
```

Si se ejecuta correctamente, este comando proporciona información similar a la siguiente.

```
{
  "ScheduledUpdateGroupActions": [
    {
      "AutoScalingGroupName": "my-asg",
      "ScheduledActionName": "my-recurring-action",
      "Recurrence": "30 0 1 1,6,12 *",
      "ScheduledActionARN": "arn:aws:autoscaling:us-
west-2:123456789012:scheduledUpdateGroupAction:8e86b655-b2e6-4410-8f29-
b4f094d6871c:autoScalingGroupName/my-asg:scheduledActionName/my-recurring-action",
      "StartTime": "2020-12-01T00:30:00Z",
      "Time": "2020-12-01T00:30:00Z",
      "MinSize": 1,
      "MaxSize": 6,
      "DesiredCapacity": 4
    }
  ]
}
```

Verificación de actividades de escalado

Para verificar las actividades de escalado asociadas con el escalado programado, consulte Verificación de una actividad de escalado para un grupo de escalado automático.

Eliminación de una acción programada

Para eliminar una acción programada, use uno de los siguientes métodos:

Console

Para eliminar una acción programada

1. Abra la consola Amazon EC2 en https://console.aws.amazon.com/ec2/y seleccione Auto Scaling Groups en el panel de navegación.
2. Seleccione el grupo de escalado automático.
3. En la pestaña Automatic scaling (Escalado automático), en Scheduled actions (Acciones programadas), seleccione una acción programada.

4. Elija Actions (Acciones), Delete (Eliminar).
5. Cuando se le indique que confirme, seleccione Yes, Delete (Sí, borrar).

AWS CLI

Use el siguiente comando delete-scheduled-action.

```
aws autoscaling delete-scheduled-action --auto-scaling-group-name my-asg \
  --scheduled-action-name my-recurring-action
```

Escalado dinámico para Amazon EC2 Auto Scaling

El escalado dinámico escala la capacidad del grupo de escalado automático a medida que se producen cambios de tráfico

Amazon EC2 Auto Scaling admite los siguientes tipos de política de escalado dinámico:

- Escalado de seguimiento de objetivos: aumenta y reduce la capacidad actual del grupo en función de una CloudWatch métrica de Amazon y un valor objetivo. Funciona de forma similar a los termostatos, que mantienen la temperatura del hogar: seleccionamos una temperatura y el termostato hace el resto.
- Step scaling (Escalado por pasos): permite aumentar o reducir la capacidad actual del grupo en función de una serie de ajustes de escalado, denominados ajustes por pasos, que variarán en función del tamaño de la interrupción de alarma.
- Simple scaling (Escalado sencillo): permite aumentar o reducir la capacidad actual del grupo en función de un único ajuste de escalado con un periodo de recuperación entre cada actividad.

Le recomendamos que utilice políticas de escalado de seguimiento de objetivo y que elija una métrica que cambie de forma inversamente proporcional al cambio en la capacidad de su grupo de escalado automático. Por lo tanto, si duplica el tamaño de su grupo de escalado automático, la métrica disminuirá en un 50 por ciento. Esto permite que los datos de las métricas activen con precisión los eventos de escalado proporcional. Se incluyen métricas como el uso promedio de la CPU o el recuento promedio de solicitudes por objetivo.

Con el seguimiento de objetivos, el grupo de escalado automático escala en proporción directa a la carga real de la aplicación. Esto significa que, además de satisfacer la necesidad inmediata de

capacidad en respuesta a los cambios de carga, una política de seguimiento de objetivos también puede adaptarse a los cambios de carga que se producen con el tiempo, por ejemplo, debido a las variaciones estacionales.

Las políticas de seguimiento de Target también eliminan la necesidad de definir manualmente CloudWatch las alarmas y los ajustes de escalado. Amazon EC2 Auto Scaling gestiona esto automáticamente en función del objetivo que establezca.

Contenido

- Funcionamiento de las políticas de escalado dinámico
- Varias políticas de escalado dinámico
- Políticas de escalado de seguimiento de destino para Amazon EC2 Auto Scaling
- Políticas de escalado sencillo y por pasos para Amazon EC2 Auto Scaling
- Recuperaciones de escalado para Amazon EC2 Auto Scaling
- Política de escalado basada en Amazon SQS
- Verificación de una actividad de escalado para un grupo de escalado automático
- Desactivación de una política de escalado para un grupo de escalado automático
- Eliminación de una política de escalado para un grupo de escalado automático
- Ejemplos de políticas de escalado para AWS CLI

Funcionamiento de las políticas de escalado dinámico

Una política de escalado dinámico indica a Amazon EC2 Auto Scaling que realice un seguimiento de una métrica CloudWatch específica y define qué acción se debe realizar cuando la alarma CloudWatch asociada está en ALARM. Las métricas que se utilizan para invocar el estado de alarma son una agregación de métricas procedentes de todas las instancias del grupo de escalado automático. (Por ejemplo, supongamos que tiene un grupo de escalado automático con dos instancias, donde una instancia tiene un 60 % de CPU y la otra tiene un 40 % de CPU. Tienen el 50 por ciento de promedio de CPU). Cuando la política está en vigor, Amazon EC2 Auto Scaling ajusta la capacidad deseada del grupo hacia arriba o hacia abajo cuando el umbral de una alarma se interrumpe.

Cuando se invoca una política de escalado, si el cálculo de capacidad produce un número fuera del rango entre el tamaño mínimo y máximo del grupo, Amazon EC2 Auto Scaling garantiza que la nueva capacidad nunca se salga de los límites de tamaño mínimo y máximo. La capacidad se

mide de una de estas dos maneras: utilizando las mismas unidades que eligió al momento de definir la capacidad deseada en términos de instancias o utilizando unidades de capacidad (si se aplican pesos de instancia).

- Ejemplo 1: un grupo de escalado automático tiene una capacidad máxima de 3, una capacidad actual de 2 y una política de escalado dinámico que agrega 3 instancias. Al invocar esta política, Amazon EC2 Auto Scaling solo agrega 1 instancia al grupo para evitar que este supere su tamaño máximo.
- Ejemplo 2: un grupo de escalado automático tiene una capacidad mínima de 2, una capacidad actual de 3 y una política de escalado dinámico que elimina 2 instancias. Al invocar esta política, Amazon EC2 Auto Scaling solo quita 1 instancia del grupo para evitar que este sea menor que su tamaño mínimo.

Cuando la capacidad deseada alcanza el límite de tamaño máximo, el escalado ascendente se detiene. Si la demanda cae y la capacidad disminuye, Amazon EC2 Auto Scaling puede volver a escalar horizontalmente.

La excepción es cuando se utilizan pesos de instancias. En este caso, Amazon EC2 Auto Scaling puede escalar horizontalmente por encima del límite de tamaño máximo, pero solo por hasta la ponderación máxima de instancia. Su intención es acercarse lo más posible a la nueva capacidad deseada, pero aun así respetar las estrategias de asignación que se han especificado para el grupo. Las estrategias de asignación determinan los tipos de instancia que se van a lanzar. Los pesos determinan cuántas unidades de capacidad aporta cada instancia a la capacidad deseada del grupo en función de su tipo de instancia.

- Ejemplo 3: un grupo de escalado automático tiene una capacidad máxima de 12, una capacidad actual de 10 y una política de escalado dinámico que agrega 5 unidades de capacidad. Los tipos de instancia tienen una de las tres ponderaciones asignadas: 1, 4 o 6. Al invocar la política, Amazon EC2 Auto Scaling elige lanzar un tipo de instancias con una ponderación de 6 en función de la estrategia de asignación. El resultado de este evento de escalado ascendente es un grupo con una capacidad deseada de 12 y una capacidad actual de 16.

Varias políticas de escalado dinámico

En la mayoría de los casos, una política de escalado de seguimiento de destino es suficiente para configurar el grupo de escalado automático para que se escale y reduzca horizontalmente de forma automática. Una política de escalado de seguimiento de destino le permite seleccionar un

resultado deseado y hacer que el grupo de escalado automático agregue y quite instancias según sea necesario para lograr ese resultado.

Para una configuración de escalado avanzada, el grupo de escalado automático puede tener más de una política de escalado. Por ejemplo, puede definir una o más políticas de escalado de seguimiento de destino, una o más políticas de escalado por pasos o ambos tipos. Esto proporciona una mayor flexibilidad para abordar diferentes situaciones.

Para ilustrar cómo se combinan varias políticas de escalado dinámico, considere una aplicación que utiliza un grupo de escalado automático y una cola de Amazon SQS para enviar solicitudes a una sola instancia EC2. Para garantizar que la aplicación funciona en niveles óptimos, existen dos políticas que controlan cuándo debe escalarse horizontalmente el grupo de escalado automático. Una es una política de seguimiento de destino que utiliza una métrica personalizada para añadir y eliminar capacidad en función del número de mensajes SQS en la cola. La otra es una política de escalado escalonado que utiliza la CloudWatch `CPUUtilization` métrica de Amazon para añadir capacidad cuando la instancia supera el 90 por ciento de uso durante un período de tiempo específico.

Cuando hay varias políticas en vigor a la vez, existe la posibilidad de que cada una de ellas pueda indicar al grupo de escalado automático que escale (o reduzca) horizontalmente al mismo tiempo. Por ejemplo, es posible que la `CPUUtilization` métrica alcance su punto máximo y supere el umbral de la CloudWatch alarma al mismo tiempo que la métrica personalizada de SQS se dispare y supere el umbral de la alarma de métrica personalizada.

Cuando se producen estas situaciones, Amazon EC2 Auto Scaling elige la política que proporciona la mayor capacidad tanto para el escalado como para la reducción horizontal. Por ejemplo, suponga que la política `CPUUtilization` lanza una instancia, mientras que la política de la cola de SQS lanza dos instancias. Si se cumple el criterio de escalado horizontal de ambas políticas al mismo tiempo, Amazon EC2 Auto Scaling da prioridad a la política de la cola de SQS. Por consiguiente, el grupo de escalado automático lanzará dos instancias.

El enfoque de dar prioridad a la política que proporciona la mayor capacidad se aplica incluso cuando las políticas utilizan criterios diferentes para la reducción horizontal. Por ejemplo, si una política termina tres instancias, otra política disminuye el número de instancias en un 25 % y el grupo tiene ocho instancias en el momento de reducir horizontalmente, Amazon EC2 Auto Scaling prioriza la política que proporciona el mayor número de instancias para el grupo. Esto da lugar a que el grupo de escalado automático termine dos instancias (25 por ciento de 8 = 2). La intención es evitar que Amazon EC2 Auto Scaling elimine demasiadas instancias.

Sin embargo, recomendamos precaución al utilizar políticas de escalado de seguimiento de destino con políticas de escalado por pasos, ya que los conflictos entre estas políticas pueden provocar un comportamiento no deseado. Por ejemplo, si la política de escalado por pasos inicia una actividad de reducción horizontal antes de que la política de seguimiento de objetivo esté lista para la reducción horizontal, la actividad de reducción horizontal no se bloqueará. Una vez completada la actividad de reducción horizontal, la política de seguimiento de objetivo podría indicar al grupo que vuelva a escalar horizontalmente.

Políticas de escalado de seguimiento de destino para Amazon EC2 Auto Scaling

Una política de escalado de seguimiento de objetivo escala automáticamente la capacidad del grupo de escalado automático en función de un valor de métrica de destino. Se adapta automáticamente a los patrones de uso particulares de sus aplicaciones individuales. Esto permite que su aplicación mantenga un rendimiento óptimo y una elevada utilización de las instancias de EC2 para una mejor rentabilidad sin intervención manual.

Con el seguimiento de objetivo, seleccione una métrica y un valor objetivo para representar el nivel ideal de utilización promedio o rendimiento para su aplicación. Amazon EC2 Auto Scaling crea y administra CloudWatch las alarmas que invocan eventos de escalado cuando la métrica se desvía del objetivo. Para ejemplificar, esto es similar a cómo un termostato mantiene una temperatura objetivo.

Por ejemplo, supongamos que tiene una aplicación web que actualmente se pone en marcha en dos instancias y desea que la utilización de CPU del grupo de escalado automático permanezca en torno al 50 % cuando cambie la carga en la aplicación. De este modo dispone de capacidad adicional para gestionar picos de tráfico sin mantener una cantidad excesiva de recursos inactivos.

Puede satisfacer esta necesidad mediante la creación de una política de escalado de seguimiento de destino que tenga como destino una utilización media de CPU del 50 por ciento. Luego, el grupo de escalado automático se escalará horizontalmente, o aumentará su capacidad, cuando la CPU supere el 50 por ciento para gestionar el aumento de carga. Se reducirá horizontalmente, o reducirá su capacidad, cuando la CPU caiga por debajo del 50 por ciento para optimizar los costos durante los periodos de baja utilización.

Temas

- Políticas de escalado de seguimiento de destino
- Elección de métricas

- Definición del valor de destino
- Definición del tiempo de preparación de la instancia
- Consideraciones
- Creación de una política de escalado de seguimiento de destino
- Cómo crear una política de seguimiento de objetivos con métricas de alta resolución para obtener una respuesta más rápida
- Creación de una política de escalado de seguimiento de destino con cálculos métricos

Políticas de escalado de seguimiento de destino

Puede tener varias políticas de escalado de seguimiento de destino juntas que le ayuden a optimizar su rendimiento, siempre que cada una de ellas use una métrica diferente. Por ejemplo, la utilización y el rendimiento pueden influir mutuamente. Cada vez que una de estas métricas cambia, normalmente implica que otras métricas también se verán afectadas. Por lo tanto, el uso de varias métricas proporciona información adicional sobre la carga a la que se encuentra el grupo de escalado automático. Esto puede ayudar a Amazon EC2 Auto Scaling a tomar decisiones mejor informadas a la hora de determinar cuánta capacidad añadir a su grupo.

La intención de Amazon EC2 Auto Scaling es priorizar siempre la disponibilidad. Escalará horizontalmente el grupo de escalado automático si cualquiera de las políticas de seguimiento de objetivo está lista para el escalado horizontal. Solo realizará una reducción horizontal si todas las políticas de seguimiento de objetivo (que tienen habilitada la reducción horizontal) estén listas para la reducción horizontal.

Elección de métricas

Puede crear políticas de escalado de seguimiento de destino con métricas predefinidas o personalizadas. Las métricas predefinidas facilitan el acceso a las métricas más utilizadas para el escalado. Las métricas personalizadas le permiten escalar en función de otras CloudWatch métricas disponibles, incluidas las métricas de alta resolución que se publican en intervalos más precisos del orden de unos pocos segundos. Puede publicar sus propias métricas de alta resolución o las que publiquen otros servicios de AWS .

Para obtener más información sobre la creación de políticas de seguimiento de objetivos con métricas de alta resolución, consulte Cómo crear una política de seguimiento de objetivos con métricas de alta resolución para obtener una respuesta más rápida.

El seguimiento de objetivos admite las siguientes métricas predefinidas:

- `ASGAverageCPUUtilization`: promedio de utilización de la CPU del grupo de Auto Scaling.
- `ASGAverageNetworkIn`: número promedio de bytes recibidos en todas las interfaces de red por el grupo de Auto Scaling.
- `ASGAverageNetworkOut`: número promedio de bytes enviados en todas las interfaces de red por el grupo de Auto Scaling.
- `ALBRequestCountPerTarget`: recuento medio de solicitudes de Application Load Balancer por destino para su grupo de Auto Scaling.

> **Important**
>
> Encontrará más información valiosa sobre las métricas de uso de la CPU, E/S de red y recuento de solicitudes de Application Load Balancer por objetivo en el tema Enumere las métricas CloudWatch disponibles para sus instancias de la Guía del usuario de Amazon EC2 y las métricas de su Application Load Balancer de CloudWatch la Guía del usuario de Application Load Balancers, respectivamente.

Puede elegir otras CloudWatch métricas disponibles o las suyas propias especificando una métrica personalizada CloudWatch . Para ver un ejemplo que especifique una especificación de métrica personalizada para una política de escalado de seguimiento de objetivos mediante la AWS CLI, consulteEjemplos de políticas de escalado para AWS CLI.

Tenga en cuenta las siguientes consideraciones al elegir una métrica:

- Recomendamos que solo utilice métricas que estén disponibles en intervalos de un minuto o menos para permitir escalar más rápido en respuesta a los cambios de uso. Las métricas que se publican a intervalos más bajos permiten que la política de seguimiento de objetivos detecte y responda más rápido a los cambios en la utilización del grupo de escalado automático.
- Si elige métricas predefinidas publicadas por Amazon EC2, como el uso de la CPU, le recomendamos que habilite el monitoreo detallado. Por defecto, todas las métricas de Amazon EC2 se publican en intervalos de cinco minutos, pero se pueden configurar a un intervalo más bajo al habilitar el monitoreo detallado. Para obtener información acerca de cómo habilitar el monitoreo detallado, consulte Configuración de la supervisión para instancias de Auto Scaling.
- No todas las métricas personalizadas funcionan para el seguimiento de destino. La métrica debe ser una métrica de utilización válida y describir el nivel de actividad de una instancia. El valor de la métrica debe aumentar o disminuir proporcionalmente al número de instancias del grupo de

escalado automático. De esta forma, los datos de las métricas se pueden utilizar para ampliar o reducir proporcionalmente el número de instancias. Por ejemplo, la utilización de la CPU de un grupo de escalado automático (es decir, la métrica `CPUUtilization` de Amazon EC2 con la dimensión de métrica `AutoScalingGroupName`) funciona si la carga del grupo de escalado automático se distribuye entre las instancias.

- Las siguientes métricas no sirven para hacer un seguimiento de destino:
  - El número de solicitudes recibidas por el balanceador de carga frente al grupo de escalado automático (es decir, la métrica Elastic Load Balancing de `RequestCount`). El número de solicitudes recibidas por el balanceador de carga no cambia en función de la utilización del grupo de escalado automático.
  - La latencia de las solicitudes del balanceador de carga (es decir, la métrica `Latency` de Elastic Load Balancing). La latencia de las solicitudes puede aumentar si lo hace la utilización, pero el cambio no tiene que ser necesariamente proporcional.
  - La métrica CloudWatch de colas de Amazon SQS. `ApproximateNumberOfMessagesVisible` El número de mensajes de una cola no tiene por qué cambiar en proporción al tamaño del grupo de escalado automático que procesa los mensajes de la cola. Sin embargo, puede que funcione una métrica personalizada que mida el número de mensajes de la cola por instancia de EC2 en el grupo de escalado automático. Para obtener más información, consulte Política de escalado basada en Amazon SQS.
- Para utilizar la métrica `ALBRequestCountPerTarget`, debe especificar el parámetro `ResourceLabel` para identificar el grupo de destino del balanceador de carga asociado a la métrica. Para ver un ejemplo que especifique el `ResourceLabel` parámetro de una política de escalado de seguimiento de objetivos mediante el AWS CLI, consulte. Ejemplos de políticas de escalado para AWS CLI
- Cuando una métrica emite valores 0 reales a CloudWatch (por ejemplo,`ALBRequestCountPerTarget`), un grupo de Auto Scaling puede escalar a 0 cuando no haya tráfico en su aplicación durante un período prolongado de tiempo. Para que el grupo de escalado automático se reduzca horizontalmente a 0 cuando no se envíen solicitudes, la capacidad mínima del grupo debe ser de 0.
- En lugar de publicar métricas nuevas para utilizarlas en su política de escalado, puede utilizar la matemáticas métricas para combinar las métricas existentes. Para obtener más información, consulte Creación de una política de escalado de seguimiento de destino con cálculos métricos.

Definición del valor de destino

Al crear una política de escalado de seguimiento de destino, debe especificar un valor de destino. El valor objetivo representa la utilización o el rendimiento promedio óptimo para el grupo de escalado automático. Para usar los recursos de manera rentable, establezca el valor objetivo lo más alto posible con un búfer razonable para aumentos inesperados de tráfico. Cuando la aplicación se escala horizontalmente de manera óptima para un flujo de tráfico normal, el valor de la métrica real debe ser igual al valor de destino, o estar justo por debajo de él.

Cuando una política de escalado se basa en el rendimiento, como el recuento de solicitudes por objetivo para un equilibrador de carga de aplicación, E/S de red u otras métricas de recuento, el valor objetivo representa el rendimiento promedio óptimo de una sola instancia, para un periodo de un minuto.

Definición del tiempo de preparación de la instancia

Puede especificar el número de segundos que se tarda en preparar una instancia recién lanzada. Hasta que finaliza el tiempo de preparación especificado, la instancia no se tiene en cuenta para las métricas agregadas de la instancia de EC2 del grupo de escalado automático.

Mientras las instancias se encuentran en el periodo de preparación, las políticas de escalado solo escalan horizontalmente si el valor de la métrica de las instancias que no están en preparación es mayor que la utilización objetivo de la política.

Si el grupo se vuelve a escalar horizontalmente, las instancias que aún estén en preparación se considerarán parte de la capacidad deseada para la siguiente actividad de escalado horizontal. La intención es realizar continuamente (pero no excesivamente) un escalado ascendente.

Mientras la actividad de escalado horizontal está en curso, todas las actividades de reducción horizontal iniciadas por las políticas de escalado se bloquean hasta que las instancias terminen de prepararse. Cuando las instancias terminen de prepararse, si se produce un evento de reducción horizontal, cualquier instancia que se encuentre actualmente en proceso de terminación se tendrá en cuenta para la capacidad actual del grupo al calcular la nueva capacidad deseada. Por lo tanto, no se eliminan más instancias del grupo de Auto Scaling que las necesarias.

Predeterminado

Si no se establece ningún valor, la política de escalado utilizará el valor predeterminado, que es el valor de la preparación de instancias por defecto definido para el grupo. Si la preparación predeterminada de instancias es nula, se revertirá al valor de recuperación predeterminado.

Recomendamos usar la preparación de instancias predeterminada para facilitar la actualización de todas las políticas de escalado cuando cambie el tiempo de preparación.

Consideraciones

Las siguientes consideraciones se aplican al trabajar con las políticas de escalado de seguimiento de destino:

- No cree, edite ni elimine las CloudWatch alarmas que se utilizan con una política de escalado y seguimiento de Target. Amazon EC2 Auto Scaling crea y administra CloudWatch las alarmas asociadas a las políticas de escalado de seguimiento de destino y puede editarlas, sustituirlas o eliminarlas cuando sea necesario para personalizar la experiencia de escalado de sus aplicaciones y sus patrones de utilización cambiantes.
- Una política de escalado de seguimiento de objetivos prioriza la disponibilidad durante los períodos de niveles de tráfico fluctuantes al reducir horizontalmente de forma más gradual cuando el tráfico disminuye. Si desea tener un mayor control, una política de escalado por pasos podría ser la mejor opción. Puede deshabilitar la reducción horizontal de una política de seguimiento de objetivo. Esto permite mantener una cantidad mínima de instancias para que las implementaciones se realicen correctamente.
- Si a la métrica le faltan puntos de datos, el estado de la CloudWatch alarma cambia a`INSUFFICIENT_DATA`. Cuando esto ocurre, Amazon EC2 Auto Scaling no puede escalar su grupo hasta que se encuentren nuevos puntos de datos.
- Si la métrica se presenta de forma dispersa por diseño, las matemáticas métricas pueden resultar útiles. Por ejemplo, para usar los valores más recientes, utilice la función `FILL(m1,REPEAT)`, donde `m1` es la métrica.
- Es posible que haya diferencias entre el valor objetivo y los puntos de datos de la métrica real. Esto se debe a que actuamos de forma conservadora redondeando hacia arriba o hacia abajo a la hora de determinar el número de instancias que se deben agregar o quitar. Esto impide que agreguemos un número insuficiente de instancias o eliminemos demasiadas. Sin embargo, en el caso de los grupos de Auto Scaling que son más pequeños y tienen menos instancias, la utilización del grupo puede parecer que está muy lejos del valor de destino.

  Por ejemplo, supongamos que se establece un valor de destino del 50 por ciento de utilización de la CPU y el grupo de escalado automático lo supera. Es posible determinar que la adición de 1,5 instancias disminuirá la utilización de la CPU hasta aproximadamente el 50 por ciento. Como no es posible agregar 1,5 instancias, redondeamos hacia arriba y añadimos dos instancias. Esto podría reducir la utilización de la CPU a un valor inferior al 50 por ciento, pero garantizaría

que la aplicación cuenta con los recursos suficientes. Del mismo modo, si determinamos que la eliminación de 0,5 instancias aumentaría la utilización de la CPU por encima del 50 %, decidiremos no reducir horizontalmente hasta que la métrica baje lo suficiente como para que la reducción horizontal no provoque oscilaciones.

En el caso de grupos de Auto Scaling más grandes con más instancias, la utilización se distribuye entre un número de instancias mayor, en cuyo caso, si se agregan o quitan instancias, la diferencia entre el valor de destino y los puntos de datos de la métrica real es menor.

- En las políticas de escalado de seguimiento de destino, se presupone que el grupo de escalado automático debe escalarse horizontalmente cuando la métrica especificada está por encima del valor de destino. Las políticas de escalado de seguimiento de destino no pueden utilizarse para escalar horizontalmente el grupo de escalado automático si la métrica está por debajo del valor de destino.

Creación de una política de escalado de seguimiento de destino

Para crear una política de escalado de seguimiento de objetivo para el grupo de escalado automático, use uno de los siguientes métodos:

Antes de empezar, confirme que su métrica preferida esté disponible en intervalos de 1 minuto (en comparación con el intervalo de 5 minutos predeterminado de las métricas de Amazon EC2).

Console

Creación de una política de escalado de seguimiento de destino para un nuevo grupo de escalado automático

1. Abra la consola Amazon EC2 en https://console.aws.amazon.com/ec2/y seleccione Auto Scaling Groups en el panel de navegación.
2. Elija Create Auto Scaling group (Crear grupo de escalado automático).
3. En los pasos 1, 2 y 3, elija las opciones que desee y continúe en el Paso 4: Configurar el tamaño del grupo y las políticas de escalado.
4. En Escalado, especifique el rango entre el que desea escalar actualizando la Capacidad deseada mínima y la Capacidad deseada máxima. Estas dos configuraciones permiten escalar dinámicamente el grupo de escalado automático. Para obtener más información, consulte Establecimiento de límites de escalado para el grupo de escalado automático.
5. En Escalado automático, elija Política de escalado de seguimiento de destino.

6. Para definir una política, haga lo siguiente:

   a. Especifique un nombre para la política.

   b. En Tipo de métrica, elija una métrica.

      Si eligió Application Load Balancer request count per target (Recuento de solicitudes de Application Load Balancer por destino), elija un grupo de destino en Target group (Grupo de destino).

   c. Especifique un valor de destino para la métrica en Target value.

   d. (Opcional) Para Preparación de instancias, actualice el valor de preparación de la instancia según sea necesario.

   e. (Opcional) Seleccione Deshabilitar la reducción horizontal para crear solo una política de escalado horizontal. De este modo, si lo desea, puede crear por separado una política de reducción horizontal de otro tipo.

7. Proceda a crear el grupo de escalado automático. La política de escalado se creará después de que se haya creado el grupo de escalado automático.

Para crear una política de escalado de seguimiento de destino para un grupo de escalado automático existente

1. Abra la consola Amazon EC2 en https://console.aws.amazon.com/ec2/y seleccione Auto Scaling Groups en el panel de navegación.
2. Seleccione la casilla situada junto al grupo de escalado automático.

   Se abre un panel dividido en la parte inferior de la página.
3. Verifique que los límites de escalado estén establecidos correctamente. Por ejemplo, si la capacidad deseada de su grupo ya tiene el tamaño máximo, necesita especificar un nuevo máximo de escalado horizontal. Para obtener más información, consulte Establecimiento de límites de escalado para el grupo de escalado automático.
4. En la pestaña Automatic scaling (Escalado automático), en Dynamic scaling policies (Políticas de escalado dinámico), elija Create dynamic scaling policy (Crear política de escalado dinámico).
5. Para definir una política, haga lo siguiente:

   a. En Tipo de política, mantenga el valor predeterminado de Escalado de seguimiento de destino.

b. Especifique un nombre para la política.

c. En Tipo de métrica, elija una métrica. Solo puede elegir un tipo de métrica. Para utilizar más de una métrica, cree varias políticas.

   Si eligió Application Load Balancer request count per target (Recuento de solicitudes de Application Load Balancer por destino), elija un grupo de destino en Target group (Grupo de destino).

d. Especifique un valor de destino para la métrica en Target value.

e. (Opcional) Para Preparación de instancias, actualice el valor de preparación de la instancia según sea necesario.

f. (Opcional) Seleccione Deshabilitar la reducción horizontal para crear solo una política de escalado horizontal. De este modo, si lo desea, puede crear por separado una política de reducción horizontal de otro tipo.

6. Seleccione Crear.

AWS CLI

Para crear una política de escalado de seguimiento de objetivo, puede utilizar el siguiente ejemplo para empezar. Reemplace cada *user input placeholder* por su propia información.

> **Note**
>
> Para obtener más ejemplos, consulte Ejemplos de políticas de escalado para AWS CLI.

Creación de una política de escalado de seguimiento de destino (AWS CLI)

1. Utilice el siguiente comando `cat` para almacenar un valor de destino para su política de escalado y una especificación de métricas predefinida en un archivo JSON llamado `config.json` en su directorio principal. A continuación, se incluye un ejemplo de configuración de seguimiento de destino que mantiene la utilización media de la CPU en un 50 por ciento.

```
$ cat ~/config.json
{
  "TargetValue": 50.0,
  "PredefinedMetricSpecification":
```

```
    {
      "PredefinedMetricType": "ASGAverageCPUUtilization"
    }
}
```

Para obtener más información, consulte la PredefinedMetricSpecificationreferencia de la API Amazon EC2 Auto Scaling.

2. Utilice el comando put-scaling-policy, junto con el archivo `config.json` creado en el paso anterior, para crear la política de escalado.

```
aws autoscaling put-scaling-policy --policy-name cpu50-target-tracking-scaling-policy \
  --auto-scaling-group-name my-asg --policy-type TargetTrackingScaling \
  --target-tracking-configuration file://config.json
```

Si se ejecuta correctamente, este comando devuelve los nombres ARNs y los nombres de las dos CloudWatch alarmas creadas en su nombre.

```
{
    "PolicyARN": "arn:aws:autoscaling:us-
west-2:123456789012:scalingPolicy:228f02c2-c665-4bfd-
aaac-8b04080bea3c:autoScalingGroupName/my-asg:policyName/cpu50-target-tracking-
scaling-policy",
    "Alarms": [
        {
            "AlarmARN": "arn:aws:cloudwatch:us-
west-2:123456789012:alarm:TargetTracking-my-asg-AlarmHigh-
fc0e4183-23ac-497e-9992-691c9980c38e",
            "AlarmName": "TargetTracking-my-asg-AlarmHigh-
fc0e4183-23ac-497e-9992-691c9980c38e"
        },
        {
            "AlarmARN": "arn:aws:cloudwatch:us-
west-2:123456789012:alarm:TargetTracking-my-asg-AlarmLow-61a39305-ed0c-47af-
bd9e-471a352ee1a2",
            "AlarmName": "TargetTracking-my-asg-AlarmLow-61a39305-ed0c-47af-
bd9e-471a352ee1a2"
        }
    ]
}
```

Cómo crear una política de seguimiento de objetivos con métricas de alta resolución para obtener una respuesta más rápida

El seguimiento de Target admite CloudWatch métricas de alta resolución con puntos de datos de segundo nivel que se publican a intervalos inferiores a un minuto. Configure políticas de seguimiento de objetivos para supervisar la utilización mediante CloudWatch métricas de alta resolución para las aplicaciones que tienen patrones de demanda volátiles, como las API de servicio a los clientes, los servicios de streaming en directo, los sitios web de comercio electrónico y el procesamiento de datos bajo demanda. Para lograr una mayor precisión a la hora de ajustar la capacidad a la demanda, el seguimiento de objetivos utilizará este monitoreo detallado para detectar y responder a los cambios en la demanda y el uso de las instancias de EC2 con mayor rapidez.

Para obtener más información sobre cómo publicar tus métricas en alta resolución, consulta Publicar métricas personalizadas en la Guía del CloudWatch usuario de Amazon. Para acceder a las métricas de EC2 y publicarlas, como el uso de la CPU en alta resolución, puede utilizar CloudWatch un agente.

Regiones de AWS

El seguimiento de objetivos mediante métricas de alta resolución está disponible en todos los países Regiones de AWS excepto en. AWS GovCloud (US) Regions

Cómo funciona la política de seguimiento de objetivos con métricas de alta resolución

Las políticas de seguimiento de objetivos se crean definiendo la métrica de la que se quiere hacer un seguimiento y el valor objetivo que se quiere mantener para la métrica. Para escalar en función de una métrica de alta resolución, especifique el nombre de la métrica y establezca el periodo de observación de la métrica en un valor inferior a 60 segundos. El intervalo mínimo actual admitido es de 10 segundos. Puede publicar su métrica a intervalos más bajos que este.

> **Note**
>
> No se admiten periodos de métricas superiores a 60 segundos.

Puede configurar el seguimiento de objetivos en una única CloudWatch métrica o consultar varias CloudWatch métricas y utilizar expresiones matemáticas para crear nuevas series temporales únicas basadas en estas métricas. Ambas opciones permiten definir el periodo de métricas.

Ejemplos

Ejemplo 1

En el siguiente ejemplo, se crea una política de seguimiento de objetivos basada en una CloudWatch métrica de alta resolución. La métrica se publica con una resolución de 10 segundos. Tras definir el periodo, puede habilitar el seguimiento de objetivos para monitorear esta métrica con una granularidad de 10 segundos. Reemplace cada *user input placeholder* por su propia información.

```
$ cat ~/config.json
{
  "TargetValue": 100.0,
  "CustomizedMetricSpecification": {
      "MetricName": "MyHighResolutionMetric",
      "Namespace": "MyNamespace",
      "Dimensions": [
        {
          "Name": "MyOptionalDimensionName",
          "Value": "MyOptionalMetricDimensionValue"
        }
      ],
      "Statistic": "Average",
      "Unit": "None"
      "Period": "10
  }
}
```

Ejemplo 2

Puede usar expresiones matemáticas métricas para combinar varias métricas en una única serie temporal para el escalado. Las matemáticas métricas son especialmente útiles para convertir las métricas existentes en un promedio por instancia. La conversión de métricas es importante porque el seguimiento de objetivos asume que la métrica es inversamente proporcional a la capacidad del grupo de escalado automático. Por lo tanto, cuando la capacidad aumenta, la métrica debería disminuir casi en la misma proporción.

Por ejemplo, suponga que tiene una métrica que representa los trabajos pendientes que debe procesar su aplicación. Puede utilizar las matemáticas métricas para dividir los trabajos pendientes entre la capacidad de ejecución de su grupo de escalado automático. Auto Scaling publica la métrica de capacidad con una granularidad de 1 minuto, por lo que no habrá ningún valor para esta métrica

en intervalos de menos de un minuto. Si desea utilizar una resolución más alta para el escalado, esto puede provocar una discordancia temporal entre la métrica de capacidad y de tareas pendientes. Para evitar este desajuste, recomendamos utilizar la expresión FILL para rellenar los valores faltantes con el número de capacidad registrado en la marca de tiempo del minuto anterior.

En el siguiente ejemplo se utilizan las matemáticas métricas para dividir la métrica de trabajos pendientes por la capacidad. En el caso del periodo, estableceremos ambas métricas en 10 segundos. Puesto que la métrica se publica a intervalos de 1 minuto, utilizaremos la operación FILL en la métrica de capacidad.

Cómo utilizar las matemáticas métricas para modificar varias métricas

```
{
    "CustomizedMetricSpecification": {
        "Metrics": [
            {
                "Label": "Pending jobs to be processed",
                "Id": "m1",
                "MetricStat": {
                    "Metric": {
                        "MetricName": "MyPendingJobsMetric",
                        "Namespace": "Custom",
                    },
                    "Stat": "Sum"
                    "Period": 10
                },
                "ReturnData": false
            },
            {
                "Label": "Get the running instance capacity (matching the period to
 that of the m1)",
                "Id": "m2",
                "MetricStat": {
                    "Metric": {
                        "MetricName": "GroupInServiceInstances",
                        "Namespace": "AWS/AutoScaling",
                        "Dimensions": [
                            {
                                "Name": "AutoScalingGroupName",
                                "Value": "my-asg"
                            }
                        ]
                    },
```

```
                    "Stat": "Average"
                    "Period": 10
                },
                "ReturnData": false
            },
            {
                "Label": "Calculate the pending job per capacity (note the use of the
 FILL expression)",
                "Id": "e1",
                "Expression": "m1 / FILL(m2,REPEAT)",
                "ReturnData": true
            }
        ]
    },
    "TargetValue": 100
}
```

Consideraciones

Tenga presente lo siguiente al utilizar el seguimiento de objetivos y las métricas de alta resolución.

- Para asegurarse de que no le faltan puntos de datos que puedan generar resultados de escalado automático no deseados, la CloudWatch métrica debe publicarse con una resolución igual o superior a la del período que especifique.
- Defina el valor objetivo como el valor per-instance-per-minute métrico que desea mantener para su grupo de Auto Scaling. Establecer un valor objetivo adecuado es fundamental si utiliza una métrica cuyo valor se pueda multiplicar según el periodo de la métrica. Por ejemplo, cualquier métrica basada en recuentos, como el recuento de solicitudes o trabajos pendientes, que utilice la estadística SUM tendrá un valor de métrica diferente según el periodo elegido. Aun así, debe asumir que está fijando un objetivo con respecto al promedio por minuto.
- Si bien no hay tarifas adicionales por el uso de Amazon EC2 Auto Scaling, debe pagar los recursos, como las CloudWatch instancias, las métricas CloudWatch y las alarmas de Amazon EC2. Las alarmas de alta resolución creadas en el ejemplo anterior tienen un precio diferente al de las alarmas estándar. CloudWatch Para obtener más información sobre CloudWatch los precios, consulta Amazon CloudWatch Pricing.
- El seguimiento de objetivos requiere que las métricas representen el uso promedio por instancia de las instancias de EC2. Para lograrlo, puede utilizar operaciones matemáticas métricas como parte de la configuración de su política de seguimiento de objetivos. Divida la métrica entre la capacidad de ejecución de su grupo de escalado automático. Asegúrese de definir el mismo periodo de

métrica para cada una de las métricas que utilice para crear una sola serie temporal. Si estas métricas se publican a intervalos diferentes, utilice la operación FILL en la métrica con el intervalo más alto para rellenar los puntos de datos faltantes.

Creación de una política de escalado de seguimiento de destino con cálculos métricos

Con la matemática métrica, puedes consultar múltiples CloudWatch métricas y usar expresiones matemáticas para crear nuevas series temporales basadas en estas métricas. Puede visualizar las series temporales resultantes en la CloudWatch consola y añadirlas a los paneles. Para obtener más información sobre las matemáticas métricas, consulte Uso de las matemáticas métricas en la Guía del CloudWatch usuario de Amazon.

Las siguientes consideraciones se aplican a las expresiones de la calculadora de métricas:

- Puede consultar cualquier CloudWatch métrica disponible. Cada métrica es una combinación única de nombre de métrica, espacio de nombres y cero o más dimensiones.
- Puede usar cualquier operador aritmético (+ - */^), función estadística (como AVG o SUM) u otra función compatible. CloudWatch
- Puede utilizar tanto las métricas como los resultados de otras expresiones matemáticas en las fórmulas de la expresión matemática.
- Todas las expresiones utilizadas en la especificación de una métrica deben devolver en última instancia una única serie temporal.
- Puede comprobar que una expresión matemática métrica es válida mediante la CloudWatch consola o la API. CloudWatch GetMetricData

Ejemplo: cola de tareas pendientes de Amazon SQS por instancia

Para calcular la cola de tareas pendientes de Amazon SQS por instancia, se toma el número aproximado de mensajes disponibles para recuperar de la cola y se divide por la capacidad de ejecución del grupo de escalado automático, que es el número de instancias con estado `InService`. Para obtener más información, consulte Política de escalado basada en Amazon SQS.

La lógica de la expresión es la siguiente:

```
sum of (number of messages in the queue)/(number of InService instances)
```

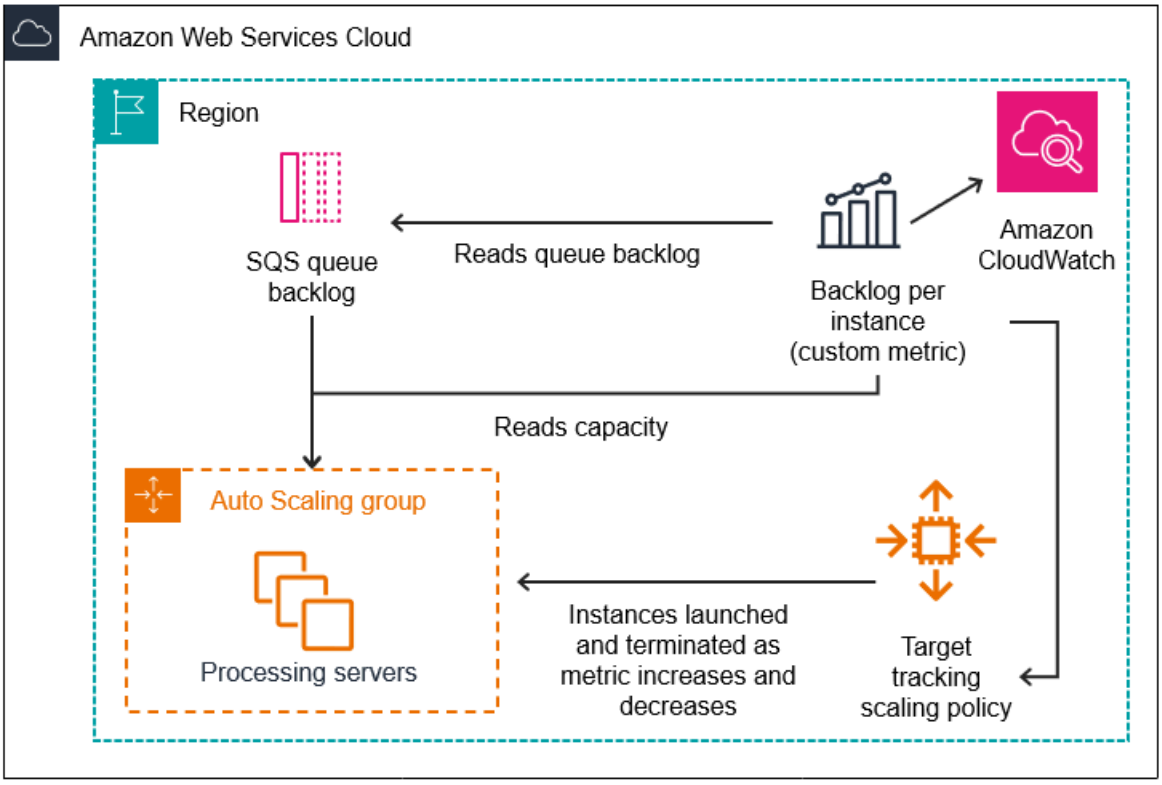
A continuación, la información de su CloudWatch métrica es la siguiente.

ID	CloudWatch métrica	Estadística	Periodo
m1	ApproximateNumberOfMessagesVisible	Sum	1 minuto
m2	GroupInServiceInstances	Media	1 minuto

Su ID de cálculo de métrica y expresión son los siguientes.

ID	Expression
e1	`(m1)/(m2)`

El siguiente diagrama ilustra la arquitectura de esta métrica:



Para utilizar esta calculadora de métricas para crear una política de escalado de seguimiento de destino (AWS CLI)

1. Guarde la expresión de la calculadora de métricas como parte de una especificación métrica personalizada en un archivo JSON denominado `config.json`.

   Utilice el siguiente ejemplo como ayuda para comenzar. Reemplace cada *`user input placeholder`* por su propia información.

```
{
    "CustomizedMetricSpecification": {
        "Metrics": [
            {
                "Label": "Get the queue size (the number of messages waiting to be
 processed)",
                "Id": "m1",
                "MetricStat": {
                    "Metric": {
                        "MetricName": "ApproximateNumberOfMessagesVisible",
                        "Namespace": "AWS/SQS",
                        "Dimensions": [
                            {
                                "Name": "QueueName",
                                "Value": "my-queue"
                            }
                        ]
                    },
                    "Stat": "Sum"
                },
                "ReturnData": false
            },
            {
                "Label": "Get the group size (the number of InService instances)",
                "Id": "m2",
                "MetricStat": {
                    "Metric": {
                        "MetricName": "GroupInServiceInstances",
                        "Namespace": "AWS/AutoScaling",
                        "Dimensions": [
                            {
                                "Name": "AutoScalingGroupName",
                                "Value": "my-asg"
                            }
```

```
                        ]
                    },
                    "Stat": "Average"
                },
                "ReturnData": false
            },
            {
                "Label": "Calculate the backlog per instance",
                "Id": "e1",
                "Expression": "m1 / m2",
                "ReturnData": true
            }
        ]
    },
    "TargetValue": 100
}
```

Para obtener más información, consulte la TargetTrackingConfigurationreferencia de la API Amazon EC2 Auto Scaling.

> **Note**
>
> Los siguientes son algunos recursos adicionales que pueden ayudarle a encontrar nombres de métricas, espacios de nombres, dimensiones y estadísticas para las métricas: CloudWatch
>
> - Para obtener información sobre las métricas disponibles para AWS los servicios, consulta AWS los servicios que publican CloudWatch métricas en la Guía del CloudWatch usuario de Amazon.
> - Para obtener el nombre, el espacio de nombres y las dimensiones exactos (si corresponde) de una CloudWatch métrica con el AWS CLI, consulta list-metrics.

2. Para crear esta política, ejecute el put-scaling-policycomando con el archivo JSON como entrada, como se muestra en el siguiente ejemplo.

```
aws autoscaling put-scaling-policy --policy-name sqs-backlog-target-tracking-scaling-policy \
  --auto-scaling-group-name my-asg --policy-type TargetTrackingScaling \
  --target-tracking-configuration file://config.json
```

Si se ejecuta correctamente, este comando devuelve el nombre de recurso de Amazon (ARN) de la política y la ARNs de las dos CloudWatch alarmas creadas en su nombre.

```
{
    "PolicyARN": "arn:aws:autoscaling:us-
west-2:123456789012:scalingPolicy:228f02c2-c665-4bfd-
aaac-8b04080bea3c:autoScalingGroupName/my-asg:policyName/sqs-backlog-target-
tracking-scaling-policy",
    "Alarms": [
        {
            "AlarmARN": "arn:aws:cloudwatch:us-
west-2:123456789012:alarm:TargetTracking-my-asg-AlarmHigh-
fc0e4183-23ac-497e-9992-691c9980c38e",
            "AlarmName": "TargetTracking-my-asg-AlarmHigh-
fc0e4183-23ac-497e-9992-691c9980c38e"
        },
        {
            "AlarmARN": "arn:aws:cloudwatch:us-
west-2:123456789012:alarm:TargetTracking-my-asg-AlarmLow-61a39305-ed0c-47af-
bd9e-471a352ee1a2",
            "AlarmName": "TargetTracking-my-asg-AlarmLow-61a39305-ed0c-47af-
bd9e-471a352ee1a2"
        }
    ]
}
```

> **Note**
>
> Si este comando arroja un error, asegúrese de haber actualizado la versión AWS CLI local a la última versión.

Políticas de escalado sencillo y por pasos para Amazon EC2 Auto Scaling

El escalado escalonado y las políticas de escalado simples escalan la capacidad de su grupo de Auto Scaling en incrementos predefinidos en función de CloudWatch las alarmas. Puede definir políticas de escalado independientes para gestionar el escalado horizontal (aumento de la capacidad) y la reducción horizontal (reducción de la capacidad) cuando se supere el umbral de una alarma.

La capacidad de los grupos de escalado automático se mide en cuanto a instancias o unidades de capacidad si se utilizan los pesos de instancia. Además, existe una diferencia entre la capacidad deseada y la capacidad actual.

- Capacidad deseada: cantidad de instancias (o unidades de capacidad) que se desea tener en el grupo. Esta se puede ajustar manual o automáticamente mediante políticas de escalado.
- Capacidad actual: cantidad de instancias (o unidades de capacidad) de su grupo que han superado los periodos de preparación y recuperación, y que están en ejecución y listas para usarse.

Con el escalado escalonado y el escalado simple, puede crear y administrar las CloudWatch alarmas que invocan el proceso de escalado. Cuando se infringe una alarma, Amazon EC2 Auto Scaling inicia la política de escalado asociada a esa alarma.

Le recomendamos que utilice políticas de escalado de seguimiento de destinos para escalar según métricas como la utilización promedio de la CPU o el recuento promedio de solicitudes por destino. Las métricas que disminuyen cuando aumenta la capacidad y aumentan cuando disminuye la capacidad se pueden usar para realizar el escalado ascendente o descendente proporcionalmente o en el número de instancias que utilizan el seguimiento de destino. Esto ayuda a garantizar que Amazon EC2 Auto Scaling siga de cerca la curva de demanda de sus aplicaciones. Para obtener más información, consulte Políticas de escalado de seguimiento de destino.

Contenido

- Cómo funcionan las políticas de escalado por pasos
- Ajustes de pasos para escalado por pasos
- Tipos de ajuste de escalado
- Preparación de las instancias
- Consideraciones
- Creación de una política de escalado por pasos de escalado horizontal
- Creación de una política de escalado por pasos de reducción horizontal
- Políticas de escalado sencillo

Cómo funcionan las políticas de escalado por pasos

Para usar el escalado por pasos, primero debe crear una CloudWatch alarma que monitoree una métrica para su grupo de Auto Scaling. Defina la métrica, el valor límite y el número de periodos de evaluación que determinan una interrupción de la alarma. A continuación, cree una política de escalado por pasos que defina cómo escalar el grupo cuando se supere el umbral de alarma. Puede utilizar un porcentaje de la capacidad actual del grupo de escalado automático o en unidades de capacidad para el tipo de ajuste de escalado. Para obtener más información, consulte Tipos de ajuste de escalado.

Añada los ajustes escalonados a la política. Puede definir diferentes ajustes escalonados en función del tamaño de la infracción de la alarma. Por ejemplo:

- Escalado horizontal en 10 instancias si la métrica de alarma alcanza el 60 por ciento
- Escalado horizontal en 30 instancias si la métrica de alarma alcanza el 75 por ciento
- Escalado horizontal en 40 instancias si la métrica de alarma alcanza el 85 por ciento

Cuando se supere el umbral de alarma durante el número especificado de periodos de evaluación, Amazon EC2 Auto Scaling aplicará los ajustes escalonados definidos en la política. Los ajustes pueden continuar en caso de que se produzcan nuevas infracciones de alarma hasta que se restablezca el estado de alarma. 0K

Cada instancia tiene un periodo de preparación para evitar que las actividades de escalado sean demasiado reactivas a los cambios que se producen en cortos periodos. Si lo desea, puede configurar el periodo de preparación para su política de escalado. Sin embargo, recomendamos usar la preparación de instancias predeterminada para facilitar la actualización de todas las políticas de escalado cuando cambie el tiempo de preparación. Para obtener más información, consulte Establecimiento de la preparación predeterminada de instancias para un grupo de escalado automático.

Las políticas de escalado simples son similares a las políticas de escalado por pasos, excepto que se basan en un único ajuste de escalado, con un periodo de recuperación entre cada actividad de escalado. Para obtener más información, consulte Políticas de escalado sencillo.

Ajustes de pasos para escalado por pasos

Cuando se crea una política de escalado por pasos, se especifican uno o varios ajustes de pasos que escalan automáticamente el número de instancias dinámicamente en función del tamaño de la interrupción de alarma. Cada ajuste por pasos especifica los elementos siguientes:

- El límite inferior del valor de la métrica
- El límite superior del valor de la métrica
- La cantidad que se va a escalar, en función del tipo de ajuste de escalado

CloudWatch agrega los puntos de datos de las métricas en función de la estadística de la métrica asociada a la alarma. CloudWatch Cuando se interrumpe la alarma, se invoca la política de escalado adecuada. Amazon EC2 Auto Scaling aplica el tipo de agregación a los puntos de datos métricos más recientes CloudWatch de (a diferencia de los datos métricos sin procesar). También compara este valor agregado de la métrica con el límite inferior y superior definido por los ajustes por pasos para determinar qué ajuste por pasos debe realizar.

Usted especifica los límites superiores e inferiores en relación con el umbral de interrupción. Por ejemplo, supongamos que ha creado una CloudWatch alarma y una política de escalado horizontal para cuando la métrica supere el 50 por ciento. A continuación, ha creado una segunda alarma y una política de reducción horizontal para cuando la métrica esté por debajo del 50 por ciento. Configuró un conjunto de ajustes de pasos con un tipo de ajuste `PercentChangeInCapacity` (o Porcentaje de grupo en la consola) para cada política:

Ejemplo: ajustes de pasos para la política de escalado ascendente

Límite inferior	Límite superior	Ajuste
0	10	0
10	20	10
20	nulo	30

Ejemplo: ajustes por pasos de la política de reducción horizontal

Límite inferior	Límite superior	Ajuste
-10	0	0
-20	-10	-10
nulo	-20	-30

Esto crea la siguiente configuración de escalado.

```
Metric value

-infinity          30%    40%          60%     70%             infinity
-------------------------------------------------------------------------
          -30%      | -10% | Unchanged  | +10%  |       +30%
-------------------------------------------------------------------------
```

Ahora, supongamos que utiliza esta configuración de escalado en un grupo de escalado automático que tiene una capacidad actual y una capacidad deseada de 10. Los siguientes puntos resumen el comportamiento de la configuración de escalado en relación con la capacidad deseada y actual del grupo:

- La capacidad deseada y actual se mantiene mientras que el valor agregado de la métrica sea superior a 40 e inferior a 60.
- Si el valor de la métrica llega a 60, la capacidad deseada del grupo aumenta en 1 instancia hasta 11 instancias, en función del segundo ajuste por pasos de la política de escalado ascendente (añadir el 10 por ciento de 10 instancias). Una vez que la nueva instancia está en ejecución y ha finalizado el tiempo de preparación especificado, la capacidad actual del grupo aumenta hasta 11 instancias. Si el valor de la métrica aumenta a 70 incluso después de este aumento de capacidad, la capacidad deseada del grupo aumenta en otras 3 instancias, hasta 14 instancias. Este valor se basa en el tercer ajuste por pasos de la política de escalado ascendente (agregar un 30% de 11 instancias, 3,3 instancias, redondeado a 3 instancias).
- Si el valor de la métrica llega a 40, la capacidad deseada del grupo se reduce en 1 instancia hasta 13 instancias, en función del segundo ajuste por pasos de la política de reducción horizontal (quitar el 10 por ciento de 14 instancias, 1,4 instancias, redondeado a 1 instancia). Si el valor de la métrica cae a 30 incluso después de esta disminución de la capacidad, la capacidad deseada del grupo disminuye en otras 3 instancias, hasta 10 instancias. Este valor se basa en el tercer ajuste por pasos de la política de reducción horizontal (quitar un 30% de 13 instancias, 3,9 instancias, redondeado a 3 instancias).

Cuando especifique los ajustes por pasos de la política de escalado, tenga en cuenta lo siguiente:

- Si usa el Consola de administración de AWS, especifica los límites superior e inferior como valores absolutos. Si utilizas el AWS CLI o un SDK, especificas los límites superior e inferior en relación con el umbral de incumplimiento.

- Los intervalos de los ajustes por pasos no se pueden solapar ni contener huecos.
- Solo puede haber un ajuste por pasos con un límite inferior nulo (infinito negativo). Si un ajuste por pasos tiene un límite inferior negativo, debe haber un ajuste por pasos con un límite inferior nulo.
- Solo puede haber un ajuste por pasos con un límite superior nulo (infinito positivo). Si un ajuste por pasos tiene un límite superior positivo, debe haber un ajuste por pasos con un límite superior nulo.
- Los límites superior e inferior no pueden ser nulos en el mismo ajuste por pasos.
- Si el valor de la métrica es superior al umbral de infracción, el límite inferior es inclusivo y el límite superior es exclusivo. Si el valor de la métrica es inferior al umbral de infracción, el límite inferior es exclusivo y el límite superior es inclusivo.

Tipos de ajuste de escalado

Puede definir una política de escalado que realice la acción de escalado idónea en función del tipo de ajuste de escalado elegido. Puede especificar el tipo de ajuste como porcentaje de la capacidad actual del grupo de escalado automático o en unidades de capacidad. Normalmente, una unidad de capacidad significa una instancia, a menos que esté utilizando la característica de pesos de instancias.

Amazon EC2 Auto Scaling admite los siguientes tipos de ajuste para el escalado sencillo y por pasos:

- `ChangeInCapacity`: permite aumentar o reducir la capacidad actual del grupo en el valor especificado. Un valor positivo aumenta la capacidad y un valor negativo reduce la capacidad. Por ejemplo: si la capacidad actual del grupo es 3 y el ajuste es 5, entonces cuando se ejecuta esta política, agregamos 5 unidades a la capacidad hasta un total de 8 unidades de capacidad.
- `ExactCapacity`: permite cambiar la capacidad actual del grupo al valor especificado. Especifique un valor positivo con este tipo de ajuste. Ejemplo: si la capacidad actual del grupo es de 3 y el ajuste es de 5, cuando se ejecute esta política, cambiamos la capacidad a 5 unidades de capacidad.
- `PercentChangeInCapacity`: permite aumentar o reducir la capacidad actual del grupo en el porcentaje especificado. Un valor positivo aumenta la capacidad y un valor negativo reduce la capacidad. Por ejemplo: si la capacidad actual es 10 y el ajuste es del 10 por ciento, entonces cuando se ejecuta esta política, agregamos 1 unidad de capacidad a la capacidad hasta un total de 11 unidades de capacidad.

> **Note**
>
> Si el valor resultante no es un entero, se redondea como se indica a continuación:
>
> - Los valores mayores que 1 se redondean al valor inferior. Por ejemplo, `12.7` se redondea a `12`.
> - Los valores comprendidos entre 0 y 1 se redondean a 1. Por ejemplo, `.67` se redondea a `1`.
> - Los valores comprendidos entre 0 y -1 se redondean a -1. Por ejemplo, `-.58` se redondea a `-1`.
> - Los valores menores que -1 se redondean al valor superior. Por ejemplo, `-6.67` se redondea a `-6`.

Con `PercentChangeInCapacity`, también puede especificar el número mínimo de instancias que escalar mediante el parámetro `MinAdjustmentMagnitude`. Suponga, por ejemplo, que crea una política que agrega un 25 por ciento y especifica un incremento mínimo de 2 instancias. Si tiene un grupo de escalado automático con 4 instancias y se ejecuta la política de escalado, el 25 por ciento de 4 es 1 instancia. Sin embargo, puesto que ha especificado un incremento mínimo de 2, se añaden 2 instancias.

Cuando se utilizan pesos de instancias, cambia el efecto de establecer el parámetro `MinAdjustmentMagnitude` en un valor distinto de cero. El valor está en unidades de capacidad. Para establecer el número mínimo de instancias que escalar, establezca este parámetro en un valor que sea al menos tan grande como la ponderación de instancia más grande.

Si utiliza pesos de instancias, tenga en cuenta que la capacidad actual del grupo de escalado automático puede exceder la capacidad deseada según sea necesario. Si el número absoluto que se va a disminuir o la cantidad que el porcentaje indica que se va a disminuir, es menor que la diferencia entre la capacidad actual y la capacidad deseada, no se ejecuta ninguna acción de escalado. Debe tener en cuenta estos comportamientos cuando observe los resultados de una política de escalado cuando se interrumpe el umbral de una alarma. Por ejemplo, supongamos que la capacidad deseada sea 30 y la capacidad actual sea 32. Cuando se interrumpe la alarma, si la política de escalado disminuye la capacidad deseada en 1, no se pone en marcha ninguna acción de escalado.

Preparación de las instancias

Para el escalado por pasos, puede especificar el número de segundos que se tarda en preparar una instancia recién lanzada. Hasta que finaliza el tiempo de preparación especificado, la instancia no se tiene en cuenta para las métricas agregadas de la instancia de EC2 del grupo de escalado automático.

Mientras las instancias se encuentran en el periodo de preparación, las políticas de escalado solo escalan horizontalmente si el valor de la métrica de las instancias que no están en preparación es mayor que el umbral superior de la alarma de la política.

Si el grupo se vuelve a escalar horizontalmente, las instancias que aún estén en preparación se considerarán parte de la capacidad deseada para la siguiente actividad de escalado horizontal. Por lo tanto, varias interrupciones de alarmas que estén en el intervalo del mismo ajuste por pasos producirán una sola actividad de escalado. La intención es realizar continuamente (pero no excesivamente) un escalado ascendente.

Por ejemplo, supongamos que crea una política de dos pasos. El primer paso agrega un 10 por ciento cuando la métrica llega a 60 y el segundo paso agrega un 30 por ciento cuando la métrica llega al 70 por ciento. La capacidad deseada y actual de su grupo de escalado automático es 10. La capacidad deseada y actual no cambia mientras que el valor agregado de la métrica es inferior a 60. Supongamos que la métrica llega a 60, por lo que se agrega 1 instancia (el 10 por ciento de las 10 instancias). Luego, la métrica pasa a 62 mientras la nueva instancia aún se está preparando. La política de escalado calcula la nueva capacidad deseada en función de la capacidad actual, que sigue siendo de 10. Sin embargo, la capacidad deseada del grupo ya se ha incrementado a 11 instancias, de modo que la política de escalado no aumenta más la capacidad deseada. Si la métrica llega a 70 mientras la nueva instancia sigue en proceso de preparación, deberíamos agregar 3 instancias (el 30 por ciento de 10 instancias). Sin embargo, la capacidad deseada del grupo es ya 11, por lo que añadimos solo 2 instancias, para una nueva capacidad deseada de 13 instancias.

Mientras la actividad de escalado horizontal está en curso, todas las actividades de reducción horizontal iniciadas por las políticas de escalado se bloquean hasta que las instancias terminen de prepararse. Cuando las instancias terminen de prepararse, si se produce un evento de reducción horizontal, cualquier instancia que se encuentre actualmente en proceso de terminación se tendrá en cuenta para la capacidad actual del grupo al calcular la nueva capacidad deseada. Por lo tanto, no se eliminan más instancias del grupo de Auto Scaling que las necesarias. Por ejemplo, mientras una instancia ya está finalizando, si una alarma está en el rango del ajuste del mismo paso que ha reducido la capacidad deseada en 1, no se pone en marcha ninguna acción de escalado.

Predeterminado

Si no se establece ningún valor, la política de escalado utilizará el valor predeterminado, que es el valor de la preparación de instancias por defecto definido para el grupo. Si la preparación predeterminada de instancias es nula, se revertirá al valor de recuperación predeterminado.

Consideraciones

Las siguientes consideraciones se aplican al trabajar con las políticas de escalado por pasos y sencillas:

- Considere si puede predecir los ajustes escalonados de la aplicación con la precisión suficiente como para utilizar la escala por pasos. Si la métrica de escalado aumenta o reduce en proporción a la capacidad del destino escalable, le recomendamos que, en su lugar, utilice una política de escalado de seguimiento de destino. Todavía tiene la opción de usar escalado por pasos como una política adicional para una configuración más avanzada. Por ejemplo, puede configurar una respuesta más agresiva cuando se alcance un determinado nivel de utilización.
- Asegúrese de elegir un margen adecuado entre los umbrales de escalado horizontal y escalado automático para evitar oscilaciones. La fluctuación es un bucle infinito de reducción horizontal y escalado horizontal. Es decir, si se realiza una acción de escalado, el valor de la métrica cambiaría e iniciaría otra acción de escalado en la dirección inversa.

Creación de una política de escalado por pasos de escalado horizontal

Para crear una política de escalado por pasos de escalado horizontal para el grupo de escalado automático, use uno de los siguientes métodos:

Console

Paso 1: Cree una CloudWatch alarma para el umbral máximo métrico

1. Abra la CloudWatch consola en https://console.aws.amazon.com/cloudwatch/.
2. De ser necesario, cambie la región. En la barra de navegación, seleccione la región en la que reside el grupo de escalado automático.
3. En el panel de navegación, elija Alarms, All alarms (Alarmas, Todas las alarmas) y, a continuación, elija Create alarm (Crear alarma).
4. Elija Seleccionar métrica.

5. En la pestaña All metrics (Todas las métricas), elija EC2, By Auto Scaling Group (Por grupo de escalado automático) y escriba el nombre del grupo de escalado automático en el campo de búsqueda. A continuación, seleccione `CPUUtilization` y elija Seleccionar métrica. Aparece la página Specify metric and conditions (Especificar métrica y condiciones), que muestra un gráfico y otra información sobre la métrica.
6. En Periodo, elija el periodo de evaluación para la alarma, por ejemplo, 1 minuto. Al evaluar la alarma, cada periodo se agrega a un punto de datos.

   > **Note**
   >
   > Un periodo más corto crea una alarma con más sensibilidad.

7. En Condiciones, haga lo siguiente:

   - En Threshold type (Tipo de umbral), elija Static (Estático).
   - En Cuando **`CPUUtilization`** sea, especifique si desea que el valor de la métrica sea mayor que o mayor o igual que el umbral para interrumpir la alarma. A continuación, en than (que), escriba el valor del umbral que desea utilizar para interrumpir la alarma.

8. En Configuración adicional, haga lo siguiente:

   - En Datapoints to alarm (Puntos de datos para la alarma), ingrese el número de puntos de datos (periodos de evaluación) durante los que el valor de la métrica debe cumplir las condiciones del umbral para interrumpir la alarma. Por ejemplo, dos periodos consecutivos de 5 minutos tardarían 10 minutos en invocar el estado de la alarma.
   - En Tratamiento de datos faltantes, elija Tratar datos faltantes como incorrectos (umbral de incumplimiento). Para obtener más información, consulta Cómo configurar el modo en que CloudWatch las alarmas tratan los datos faltantes en la Guía del CloudWatch usuario de Amazon.

9. Elija Siguiente.

   La página Configure actions (Configurar acciones) aparecerá.

10. En Notification (Notificación), seleccione el tema de Amazon SNS al que desee enviar la notificación cuando la alarma tenga el estado `ALARM`, `OK` o `INSUFFICIENT_DATA`.

    Para que la alarma envíe varias notificaciones para el mismo estado de alarma o para estados de alarma diferentes, seleccione Add notificación (Añadir notificación).

Para que la alarma no envíe notificaciones, elija Remove (Eliminar).

11. Puede dejar el resto de secciones de la página Configure actions (Configurar acciones) vacía. Si se dejan las demás secciones vacías, se crea una alarma sin asociarla a una política de escalado. A continuación, puede asociar la alarma con una política de escalado desde la consola de Amazon EC2 Auto Scaling.
12. Elija Siguiente.
13. Escriba un nombre (por ejemplo, `Step-Scaling-AlarmHigh-AddCapacity`) y, si quiere, una descripción de la alarma y, a continuación, elija Next (Siguiente).
14. Elija Crear alarma.

Utilice el siguiente procedimiento para continuar donde lo dejó después de crear la CloudWatch alarma.

Paso 2: cree de una política de escalado por pasos para escalado horizontal

1. Abra la consola Amazon EC2 en https://console.aws.amazon.com/ec2/y seleccione Auto Scaling Groups en el panel de navegación.
2. Seleccione la casilla situada junto al grupo de escalado automático.

   Se abre un panel dividido en la parte inferior de la página.
3. Verifique que los límites de escalado estén establecidos correctamente. Por ejemplo, si la capacidad deseada de su grupo ya tiene el tamaño máximo, necesita especificar un nuevo máximo de escalado horizontal. Para obtener más información, consulte Establecimiento de límites de escalado para el grupo de escalado automático.
4. En la pestaña Automatic scaling (Escalado automático), en Dynamic scaling policies (Políticas de escalado dinámico), elija Create dynamic scaling policy (Crear política de escalado dinámico).
5. En Tipo de política, elija Escalado por pasos y, a continuación, especifique un nombre para la política.
6. Para la CloudWatch alarma, elija la suya. Si aún no ha creado una alarma, elija Crear una CloudWatch alarma y complete los pasos 4 a 14 del procedimiento anterior para crear una alarma.
7. Especifique el cambio en el tamaño de grupo actual que hará esta política cuando se ejecute utilizando Take the action (Realizar la acción). Puede agregar un número específico de

instancias o un porcentaje del tamaño de grupo existente, o establecer el grupo en un tamaño exacto.

Por ejemplo, para crear una política de escalado horizontal que aumente la capacidad del grupo en un 30 por ciento, elija Add, ingrese `30` en el siguiente campo y, a continuación, elija `percent of group`. De forma predeterminada, el límite inferior de este ajuste por pasos es el límite de alarma y el límite superior es infinito positivo (+).

8. Para agregar otro paso, elija Add step (Agregar paso) y, a continuación, defina la cantidad por la que se va a escalar y los límites inferior y superior del paso en relación con el umbral de alarma.
9. Para establecer un número mínimo de instancias que escalar, actualice el campo numérico en Add capacity units in increments of at least (Agregar unidades de capacidad en incrementos de al menos) 1 capacity units (unidades de capacidad).
10. (Opcional) Para Preparación de instancias, actualice el valor de preparación de la instancia según sea necesario.
11. Seleccione Crear.

AWS CLI

Para crear una política de escalado por pasos para escalar horizontalmente (aumentar la capacidad), puede utilizar los siguientes comandos de ejemplo. Reemplace cada *`user input placeholder`* por su propia información.

Al utilizar la AWS CLI, primero debe crear una política de escalado escalonado que proporciona instrucciones a Amazon EC2 Auto Scaling sobre cómo escalar de forma horizontal cuando el valor de una métrica aumenta. Luego, cree la alarma e identifique la métrica que desea vigilar, defina las condiciones del umbral de métrica alto y otros detalles para las alarmas, y asocie la alarma a la política de escalado.

Paso 1: cree una política de escalado horizontal

Utilice el siguiente put-scaling-policycomando para crear una política de escalado escalonado denominada`my-step-scale-out-policy`, con un tipo de ajuste `PercentChangeInCapacity` que aumente la capacidad del grupo en función de los siguientes ajustes escalonados (suponiendo un umbral de CloudWatch alarma del 60 por ciento):

- Aumente el recuento de instancias en un 10 por ciento cuando el valor de la métrica sea mayor o igual al 60 por ciento pero inferior al 75 por ciento

- Aumente el recuento de instancias en un 20 por ciento cuando el valor de la métrica sea mayor o igual al 75 por ciento pero inferior al 85 por ciento
- Aumente el recuento de instancias en un 30 por ciento cuando el valor de la métrica sea mayor o igual al 85 por ciento

```
aws autoscaling put-scaling-policy \
  --auto-scaling-group-name my-asg  \
  --policy-name my-step-scale-out-policy \
  --policy-type StepScaling \
  --adjustment-type PercentChangeInCapacity \
  --metric-aggregation-type Average \
  --step-adjustments
 MetricIntervalLowerBound=0.0,MetricIntervalUpperBound=15.0,ScalingAdjustment=10 \

 MetricIntervalLowerBound=15.0,MetricIntervalUpperBound=25.0,ScalingAdjustment=20 \
                     MetricIntervalLowerBound=25.0,ScalingAdjustment=30 \
  --min-adjustment-magnitude 1
```

Registre el nombre de recurso de Amazon (ARN) de la política. Lo necesita para crear una CloudWatch alarma para la política.

```
{
    "PolicyARN":
 "arn:aws:autoscaling:region:123456789012:scalingPolicy:4ee9e543-86b5-4121-b53b-
aa4c23b5bbcc:autoScalingGroupName/my-asg:policyName/my-step-scale-in-policy
}
```

Paso 2: Cree una CloudWatch alarma para el umbral métrico alto

Utilice el siguiente CloudWatch put-metric-alarmcomando para crear una alarma que aumente el tamaño del grupo de Auto Scaling en función de un valor umbral de CPU promedio del 60 por ciento durante al menos dos períodos de evaluación consecutivos de dos minutos. Para usar su propia métrica personalizada, especifique su nombre en `--metric-name` y su espacio de nombres en `--namespace`.

```
aws cloudwatch put-metric-alarm --alarm-name Step-Scaling-AlarmHigh-AddCapacity \
  --metric-name CPUUtilization --namespace AWS/EC2 --statistic Average \
  --period 120 --evaluation-periods 2 --threshold 60 \
  --comparison-operator GreaterThanOrEqualToThreshold \
```

```
--dimensions "Name=AutoScalingGroupName,Value=my-asg" \
--alarm-actions PolicyARN
```

Creación de una política de escalado por pasos de reducción horizontal

Para crear una política de escalado por pasos de reducción horizontal para el grupo de escalado automático, use uno de los siguientes métodos:

Console

Paso 1: Cree una CloudWatch alarma para el umbral métrico bajo

1. Abra la CloudWatch consola en https://console.aws.amazon.com/cloudwatch/.
2. De ser necesario, cambie la región. En la barra de navegación, seleccione la región en la que reside el grupo de escalado automático.
3. En el panel de navegación, elija Alarms, All alarms (Alarmas, Todas las alarmas) y, a continuación, elija Create alarm (Crear alarma).
4. Elija Seleccionar métrica.
5. En la pestaña All metrics (Todas las métricas), elija EC2, By Auto Scaling Group (Por grupo de escalado automático) y escriba el nombre del grupo de escalado automático en el campo de búsqueda. A continuación, seleccione `CPUUtilization` y elija Seleccionar métrica. Aparece la página Specify metric and conditions (Especificar métrica y condiciones), que muestra un gráfico y otra información sobre la métrica.
6. En Periodo, elija el periodo de evaluación para la alarma, por ejemplo, 1 minuto. Al evaluar la alarma, cada periodo se agrega a un punto de datos.

   > **Note**
   >
   > Un periodo más corto crea una alarma con más sensibilidad.

7. En Condiciones, haga lo siguiente:

   - En Threshold type (Tipo de umbral), elija Static (Estático).
   - En Cuando **`CPUUtilization`** sea, especifique si desea que el valor de la métrica sea menor o menor o igual que el umbral para interrumpir la alarma. A continuación, en than (que), escriba el valor del umbral que desea utilizar para interrumpir la alarma.

> **Important**
>
> Para que una alarma se utilice con una política de reducción horizontal (métrica baja), asegúrese de no elegir mayor o mayor o igual que el umbral.

8. En Configuración adicional, haga lo siguiente:
   - En Datapoints to alarm (Puntos de datos para la alarma), ingrese el número de puntos de datos (periodos de evaluación) durante los que el valor de la métrica debe cumplir las condiciones del umbral para interrumpir la alarma. Por ejemplo, dos periodos consecutivos de 5 minutos tardarían 10 minutos en invocar el estado de la alarma.
   - En Tratamiento de datos faltantes, elija Tratar datos faltantes como incorrectos (umbral de incumplimiento). Para obtener más información, consulta Cómo configurar el modo en que CloudWatch las alarmas tratan los datos faltantes en la Guía del CloudWatch usuario de Amazon.
9. Elija Siguiente.

   La página Configure actions (Configurar acciones) aparecerá.
10. En Notification (Notificación), seleccione el tema de Amazon SNS al que desee enviar la notificación cuando la alarma tenga el estado `ALARM`, `OK` o `INSUFFICIENT_DATA`.

    Para que la alarma envíe varias notificaciones para el mismo estado de alarma o para estados de alarma diferentes, seleccione Add notificación (Añadir notificación).

    Para que la alarma no envíe notificaciones, elija Remove (Eliminar).
11. Puede dejar el resto de secciones de la página Configure actions (Configurar acciones) vacía. Si se dejan las demás secciones vacías, se crea una alarma sin asociarla a una política de escalado. A continuación, puede asociar la alarma con una política de escalado desde la consola de Amazon EC2 Auto Scaling.
12. Elija Siguiente.
13. Escriba un nombre (por ejemplo, `Step-Scaling-AlarmLow-RemoveCapacity`) y, si quiere, una descripción de la alarma y, a continuación, elija Next (Siguiente).
14. Elija Crear alarma.

Utilice el siguiente procedimiento para continuar donde lo dejó después de crear la CloudWatch alarma.

Paso 2: cree una política de escalado por pasos para reducción horizontal

1. Abra la consola Amazon EC2 en https://console.aws.amazon.com/ec2/y seleccione Auto Scaling Groups en el panel de navegación.
2. Seleccione la casilla situada junto al grupo de escalado automático.

   Se abre un panel dividido en la parte inferior de la página.
3. Verifique que los límites de escalado estén establecidos correctamente. Por ejemplo, si la capacidad deseada de su grupo ya tiene el tamaño mínimo, necesita especificar un nuevo mínimo de reducción horizontal. Para obtener más información, consulte Establecimiento de límites de escalado para el grupo de escalado automático.
4. En la pestaña Automatic scaling (Escalado automático), en Dynamic scaling policies (Políticas de escalado dinámico), elija Create dynamic scaling policy (Crear política de escalado dinámico).
5. En Tipo de política, elija Escalado por pasos y, a continuación, especifique un nombre para la política.
6. Para la CloudWatch alarma, elija la suya. Si aún no ha creado una alarma, elija Crear una CloudWatch alarma y complete los pasos 4 a 14 del procedimiento anterior para crear una alarma.
7. Especifique el cambio en el tamaño de grupo actual que hará esta política cuando se ejecute utilizando Take the action (Realizar la acción). Puede eliminar un número específico de instancias o un porcentaje del tamaño de grupo existente, o establecer el grupo en un tamaño exacto.

   Por ejemplo, para crear una política de reducción horizontal que reduzca la capacidad del grupo en dos instancias, elija `Remove`, introduzca 2 en el siguiente campo y, a continuación, elija `capacity units`. De forma predeterminada, el límite superior de este ajuste por pasos es el límite de alarma y el límite inferior es infinito negativo (-).
8. Para agregar otro paso, elija Add step (Agregar paso) y, a continuación, defina la cantidad por la que se va a escalar y los límites inferior y superior del paso en relación con el umbral de alarma.
9. Seleccione Crear.

AWS CLI

Para crear una política de escalado por pasos para reducir horizontalmente (disminuir la capacidad), puede utilizar los siguientes comandos de ejemplo. Reemplace cada *user input placeholder* por su propia información.

Al utilizar la AWS CLI, primero debe crear una política de escalado escalonado que proporciona instrucciones a Amazon EC2 Auto Scaling sobre cómo escalar cuando el valor de una métrica disminuye. Luego, cree la alarma e identifique la métrica que desea vigilar, defina las condiciones del umbral de métrica bajo y otros detalles para las alarmas, y asocie la alarma a la política de escalado.

Paso 1: cree una política de reducción horizontal

Utilice el siguiente put-scaling-policycomando para crear una política de escalado escalonado denominadamy-step-scale-in-policy, con un tipo de ajuste ChangeInCapacity que reduzca la capacidad del grupo en 2 instancias cuando la CloudWatch alarma asociada supere el valor mínimo métrico del umbral.

```
aws autoscaling put-scaling-policy \
  --auto-scaling-group-name my-asg  \
  --policy-name my-step-scale-in-policy \
  --policy-type StepScaling \
  --adjustment-type ChangeInCapacity \
  --step-adjustments MetricIntervalUpperBound=0.0,ScalingAdjustment=-2
```

Registre el nombre de recurso de Amazon (ARN) de la política. Lo necesita para crear la CloudWatch alarma de la política.

```
{
    "PolicyARN": "arn:aws:autoscaling:region:123456789012:scalingPolicy:ac542982-
cbeb-4294-891c-a5a941dfa787:autoScalingGroupName/my-asg:policyName/my-step-scale-
out-policy
}
```

Paso 2: Cree una CloudWatch alarma para el umbral métrico bajo

Use el siguiente CloudWatch put-metric-alarmcomando para crear una alarma que reduzca el tamaño del grupo de Auto Scaling en función del valor umbral promedio de la CPU del 40 por ciento durante al menos dos períodos de evaluación consecutivos de dos minutos. Para usar

su propia métrica personalizada, especifique su nombre en `--metric-name` y su espacio de nombres en `--namespace`.

```
aws cloudwatch put-metric-alarm --alarm-name Step-Scaling-AlarmLow-RemoveCapacity \
  --metric-name CPUUtilization --namespace AWS/EC2 --statistic Average \
  --period 120 --evaluation-periods 2 --threshold 40 \
  --comparison-operator LessThanOrEqualToThreshold \
  --dimensions "Name=AutoScalingGroupName,Value=my-asg" \
  --alarm-actions PolicyARN
```

Políticas de escalado sencillo

En los siguientes ejemplos, se muestra cómo puede utilizar comandos de la CLI para crear políticas de escalado simples. Permanecen en este documento como referencia para cualquier cliente que quiera usarlos, pero le recomendamos que utilice políticas de seguimiento de objetivo o escalado por pasos.

Al igual que las políticas de escalado escalonado, las políticas de escalado simples requieren que cree CloudWatch alarmas para sus políticas de escalado. En las políticas que cree, debe definir si desea añadir o quitar instancias, y cuántas, o establecer de manera exacta el tamaño del grupo.

Una de las principales diferencias entre las políticas de escalado por pasos y las políticas de escalado simples son los ajustes escalonados que se obtienen con las políticas de escalado por pasos. Con el escalado por pasos, puede realizar cambios mayores o menores en el tamaño del grupo en función de los ajustes escalonados que especifique.

Una política de escalado simple también debe esperar a que se complete una actividad de escalado en curso o una sustitución de comprobación de estado y a que finalice un periodo de recuperación para responder a las alarmas adicionales. Por el contrario, con el escalado por pasos, la política continúa respondiendo a alarmas adicionales, incluso mientras se está realizando una actividad de escalado o una sustitución de comprobación de estado. Esto significa que Amazon EC2 Auto Scaling evalúa todas las interrupciones de alarma cuando recibe los mensajes de alarma. Por esta razón, le recomendamos que utilice políticas de escalado por pasos en lugar de políticas de escalado simples aunque tenga un ajuste de un solo paso.

Al principio, Amazon EC2 Auto Scaling únicamente admitía las políticas de escalado sencillo. Si creó su política de escalado por pasos antes de que se incorporaran las políticas de seguimiento de objetivo y las políticas por pasos, se tratará como si fuera una política de escalado sencillo.

Creación de una política de escalado simple de escalado horizontal

Utilice el siguiente put-scaling-policycomando para crear una política de escalado sencilla denominadamy-simple-scale-out-policy, con un tipo de ajuste PercentChangeInCapacity que aumente la capacidad del grupo en un 30 por ciento cuando la CloudWatch alarma asociada supere el valor máximo de la métrica.

```
aws autoscaling put-scaling-policy --policy-name my-simple-scale-out-policy \
  --auto-scaling-group-name my-asg --scaling-adjustment 30 \
  --adjustment-type PercentChangeInCapacity
```

Registre el nombre de recurso de Amazon (ARN) de la política. Lo necesita para crear la CloudWatch alarma de la política.

Creación de una política de escalado simple de reducción horizontal

Utilice el siguiente put-scaling-policycomando para crear una política de escalado simple denominadamy-simple-scale-in-policy, con un tipo de ajuste ChangeInCapacity que reduzca la capacidad del grupo en una instancia cuando la CloudWatch alarma asociada supere el valor mínimo métrico del umbral.

```
aws autoscaling put-scaling-policy --policy-name my-simple-scale-in-policy \
  --auto-scaling-group-name my-asg --scaling-adjustment -1 \
  --adjustment-type ChangeInCapacity --cooldown 180
```

Registre el nombre de recurso de Amazon (ARN) de la política. Lo necesita para crear la CloudWatch alarma de la política.

Recuperaciones de escalado para Amazon EC2 Auto Scaling

Important

Como práctica recomendada, le recomendamos que no utilice políticas de escalado sencillo ni recuperación de escalado. Una política de escalado de seguimiento de objetivo o una política de escalado por pasos son mejores para escalar el rendimiento. Para una política de escalado que cambie el tamaño del grupo proporcionalmente a medida que el valor de la métrica de escalado disminuya o aumente, recomendamos el seguimiento de destino en lugar del escalado sencillo o por pasos.

Cuando cree políticas de escalado simples para su grupo de escalado automático, le recomendamos que configure la recuperación de escalado al mismo tiempo.

Una vez que el grupo de escalado automático lanza o termina las instancias, espera a que finalice el periodo de recuperación antes de que otras actividades de escalado iniciadas por políticas de escalado sencillo puedan iniciarse. La intención del periodo de recuperación es permitir que un grupo de escalado automático se estabilice o deje de lanzar o terminar instancias adicionales antes de que los efectos de las actividades anteriores de escalado sean visibles.

Por ejemplo, suponga que una política de escalado sencillo para la utilización de la CPU recomienda lanzar dos instancias. Amazon EC2 Auto Scaling lanza dos instancias y, a continuación, pausa las actividades de escalado hasta que finaliza el periodo de recuperación. Una vez finalizado el periodo de recuperación, las actividades de escalado iniciadas por las políticas de escalado sencillo pueden reanudarse. Si la utilización de la CPU vuelve a superar el umbral máximo de alarma, el grupo de escalado automático se vuelve a escalar horizontalmente, y vuelve a aplicarse el periodo de recuperación. Sin embargo, si dos instancias fueron suficientes para reducir el valor de la métrica, el grupo mantendrá su tamaño actual.

Contenido

- Consideraciones
- Los enlaces de ciclo de vida pueden provocar retrasos adicionales
- Cambio del periodo de recuperación predeterminado
- Establecimiento de un periodo de recuperación para políticas de escalado sencillo específicas

Consideraciones

Las siguientes consideraciones se aplican cuando se trabaja con políticas de escalado sencillo y tiempos de recuperación de escalado:

- Las políticas de seguimiento de destino y de escalado por pasos pueden iniciar una actividad de escalado horizontal inmediatamente sin esperar a que finalice el periodo de recuperación. En cambio, cada vez que el grupo de escalado automático lanza instancias, las instancias individuales tienen un periodo de preparación. Para obtener más información, consulte Establecimiento de la preparación predeterminada de instancias para un grupo de escalado automático.
- Mientras la actividad de escalado horizontal está en curso, se bloquearán todas las actividades de reducción horizontal de pasos y de seguimiento de objetivos hasta que las instancias terminen de

prepararse. Las actividades de reducción horizontal también se pueden retrasar cuando un grupo de escalado automático se encuentra en un periodo de recuperación.

- Cuando una acción programada se inicia a la hora programada, puede desencadenar también una actividad de escalado inmediatamente sin esperar a que finalice el periodo de recuperación.
- Si una instancia adopta un estado incorrecto, Amazon EC2 Auto Scaling no espera a que se complete el periodo de recuperación antes de reemplazar la instancia en estado incorrecto.
- Cuando se lanzan o terminan varias instancias, el periodo de recuperación (tanto el predeterminado como el específico de la política de escalado) surte efecto a partir del momento en que la última instancia finaliza el lanzamiento o la terminación.
- Cuando escala manualmente el grupo de escalado automático, el comportamiento predeterminado es no esperar a que finalice el periodo de recuperación. Sin embargo, puedes anular este comportamiento y respetar el tiempo de reutilización predeterminado cuando utilices el SDK AWS CLI o un SDK para escalar manualmente.
- De forma predeterminada, Elastic Load Balancing espera 300 segundos para completar el proceso de anulación del registro (drenaje de conexión). Si el grupo está detrás de un equilibrador de carga de Elastic Load Balancing, esperará a que se anule el registro de las instancias que están terminando para poder comenzar el periodo de recuperación.

Los enlaces de ciclo de vida pueden provocar retrasos adicionales

Si se invoca un enlace de ciclo de vida, el periodo de recuperación comienza una vez que completa la acción del ciclo de vida o después de que se agote el periodo de espera. Por ejemplo, suponga que tiene un grupo de escalado automático con un enlace de ciclo de vida para el lanzamiento de instancias. Cuando la aplicación experimenta un aumento en la demanda, el grupo lanza una instancia para sumar capacidad. Como hay un enlace de ciclo de vida, la instancia se pone en estado de espera y se detienen las actividades de escalado debido a las políticas de escalado sencillo. Cuando la instancia pasa a tener el estado `InService`, se inicia el periodo de recuperación. Cuando finaliza el periodo de recuperación, se reanudan las actividades de las políticas de escalado sencillo.

Cuando Elastic Load Balancing está habilitado, a los efectos de la reducción horizontal, el periodo de recuperación se inicia cuando la instancia que se quiere terminar comienza con el drenaje de conexión (retardo de anulación del registro). El período de recuperación no espera a que finalice el drenaje de conexión o a que el enlace de ciclo de vida complete su acción. Esto significa que cualquier actividad de escalado debido a las políticas de escalado sencillo puede reanudarse tan pronto como el resultado del evento de reducción horizontal se vea reflejado en la capacidad del

grupo. De lo contrario, esperar a que se completen las tres actividades —(drenaje de conexión, un enlace de ciclo de vida y un periodo de recuperación) aumentaría significativamente la cantidad de tiempo que el grupo de escalado automático necesitaría para pausar el escalado.

Cambio del periodo de recuperación predeterminado

No puede establecer el periodo de recuperación predeterminado cuando crea inicialmente un grupo de escalado automático en la consola de Amazon EC2 Auto Scaling. De forma predeterminada, este periodo de recuperación se establece en 300 segundos (5 minutos). Si es necesario, puede actualizarlo después de crear el grupo.

Para cambiar el periodo de recuperación predeterminado (consola)

Después de crear el grupo de escalado automático, en la pestaña Details (Detalles), elija Advanced configurations (Configuraciones avanzadas) y Edit (Editar). En Default cooldown (Recuperación predeterminada), elija el periodo que desea en función de la hora de inicio de las instancias u otras necesidades de la aplicación.

Para cambiar el periodo de recuperación predeterminado (AWS CLI)

Utilice los siguientes comandos para cambiar el periodo de recuperación predeterminado para grupos de Auto Scaling nuevos o existentes. Si no se define un periodo de recuperación predeterminado, se utiliza el valor predeterminado de 300 segundos.

- create-auto-scaling-group
- update-auto-scaling-group

Para confirmar el valor del enfriamiento predeterminado, usa el comando. describe-auto-scaling-groups

Establecimiento de un periodo de recuperación para políticas de escalado sencillo específicas

De forma predeterminada, todas las políticas de escalado sencillo utilizan el periodo de recuperación predeterminado que se ha definido para el grupo de escalado automático. Para especificar un periodo de recuperación para políticas de escalado sencillo específicas, utilice el parámetro de recuperación opcional al crear o actualizar la política. Cuando se especifica un periodo de recuperación para una política, este reemplaza el periodo de recuperación predeterminado.

Un uso común de un periodo de recuperación específico de la política de escalado es con una política de reducción horizontal. Como esta política termina instancias, Amazon EC2 Auto Scaling necesita menos tiempo para determinar si debe terminar instancias adicionales. La terminación de instancias debe ser una operación mucho más rápida que el lanzamiento de instancias. Por lo tanto, el periodo de recuperación predeterminado de 300 segundos es demasiado largo. En este caso, un periodo de recuperación específico de la política de escalado con un valor inferior para la política de reducción horizontal puede ayudarlo a reducir los costos al permitir que el grupo se reduzca horizontalmente de manera más rápida.

Para crear o actualizar políticas de escalado sencillo en la consola, elija la pestaña Automatic scaling (Escalado automático) después de crear el grupo. Para crear o actualizar políticas de escalado sencillas mediante el AWS CLI, utilice el put-scaling-policycomando. Para obtener más información, consulte Políticas de escalado sencillo y por pasos.

Política de escalado basada en Amazon SQS

> **Important**
>
> La información y los pasos siguientes le muestran cómo calcular la acumulación de colas de Amazon SQS por instancia utilizando el atributo `ApproximateNumberOfMessages` queue antes de publicarlo como una métrica personalizada en. CloudWatch Sin embargo, ahora puede ahorrar el costo y el esfuerzo dedicados a publicar su propia métrica mediante la calculadora de métricas. Para obtener más información, consulte Creación de una política de escalado de seguimiento de destino con cálculos métricos.

Puede escalar el grupo de escalado automático en respuesta a los cambios en la carga del sistema en una cola de Amazon Simple Queue Service (Amazon SQS). Para obtener más información acerca del uso de Amazon SQS, consulte la Guía del desarrollador de Amazon Simple Queue Service.

Hay algunas situaciones en las que es posible que tenga que pensar en la reducción horizontal en respuesta a la actividad en una cola de Amazon SQS. Por ejemplo, supongamos que tiene una aplicación web que permite a los usuarios cargar imágenes y utilizarlas en línea. En este escenario, cada imagen requiere cambiar el tamaño y la codificación antes de poder publicarla. La aplicación se ejecuta en instancias EC2 de un grupo de escalado automático y está configurada para gestionar las velocidades típicas de carga. Las instancias con error se terminan y se sustituyen para mantener los niveles de instancia actuales en todo momento. La aplicación coloca los datos de mapa de bits sin procesar de las imágenes en una cola de SQS para su procesamiento. Procesa las imágenes y,

a continuación, publica las imágenes procesadas donde puedan verlas los usuarios. La arquitectura de este escenario funciona bien si el número de operaciones de carga de imágenes no varía con el tiempo. Sin embargo, si el número de operaciones de carga cambia con el tiempo, podría considerar la posibilidad de utilizar el escalado dinámico para escalar la capacidad del grupo de escalado automático.

Contenido

- Uso del seguimiento de destino con la métrica correcta
- Limitaciones
- Configuración del escalado basada en Amazon SQS
- Amazon SQS y protección de la reducción horizontal

Uso del seguimiento de destino con la métrica correcta

Si utiliza una política de escalado de seguimiento de destino basada en una métrica de cola de Amazon SQS personalizada, el escalado dinámico puede ajustarse a la curva de demanda de la aplicación de forma más eficaz. Para obtener más información sobre cómo elegir métricas para el seguimiento de destino, consulte Elección de métricas.

El problema de usar una métrica de CloudWatch Amazon SQS, como `ApproximateNumberOfMessagesVisible` para el seguimiento de destinos, es que es posible que el número de mensajes de la cola no cambie proporcionalmente al tamaño del grupo de Auto Scaling que procesa los mensajes de la cola. Eso se debe a que el número de mensajes de su cola de SQS no es el único factor que define el número de instancias necesarias. El número de instancias del grupo de escalado automático puede depender de varios factores, como el tiempo que se tarda en procesar un mensaje y la cantidad de latencia (retraso de la cola) aceptable.

La solución es utilizar una métrica de tareas pendientes por instancia con el valor de destino igual a las tareas pendientes aceptables por instancia que desea mantener. Puede calcular estos números como se indica a continuación:

- Lista de tareas pendientes por instancia: para determinar su lista de tareas pendientes por instancia, comience con el atributo de cola `ApproximateNumberOfMessages` para determinar la longitud de la cola de SQS (número de mensajes disponibles para recuperación de la cola). Divida ese número por la capacidad de ejecución de la flota, que para un grupo de escalado automático es el número de instancias con el estado `InService`, para obtener las tareas pendientes por instancia.

- Lista de tareas pendientes aceptables por instancia: para calcular su valor de destino, determine en primer lugar lo que la aplicación puede aceptar en términos de latencia. A continuación, deberá tomar el valor de latencia aceptable y dividirlo por el tiempo medio que una instancia EC2 tarda en procesar un mensaje.

Por ejemplo, supongamos que actualmente tiene un grupo de escalado automático con 10 instancias y el número de mensajes visibles en la cola (`ApproximateNumberOfMessages`) es 1500. Si el tiempo de procesamiento medio es de 0,1 segundos para cada mensaje y la mayor latencia aceptable es de 10 segundos, el número de tareas pendientes aceptables por instancia es de 10/0,1, que equivale a 100 mensajes. Esto significa que 100 es el valor de destino para su política de seguimiento de destino. Cuando la lista de tareas pendientes por instancia alcance el valor objetivo, se producirá un evento de escalado horizontal. Si la lista de tareas pendientes por instancia se encuentra actualmente en 150 (1500/10 instancias), su grupo se escala horizontalmente en 5 instancias para mantener la proporción con el valor de destino.

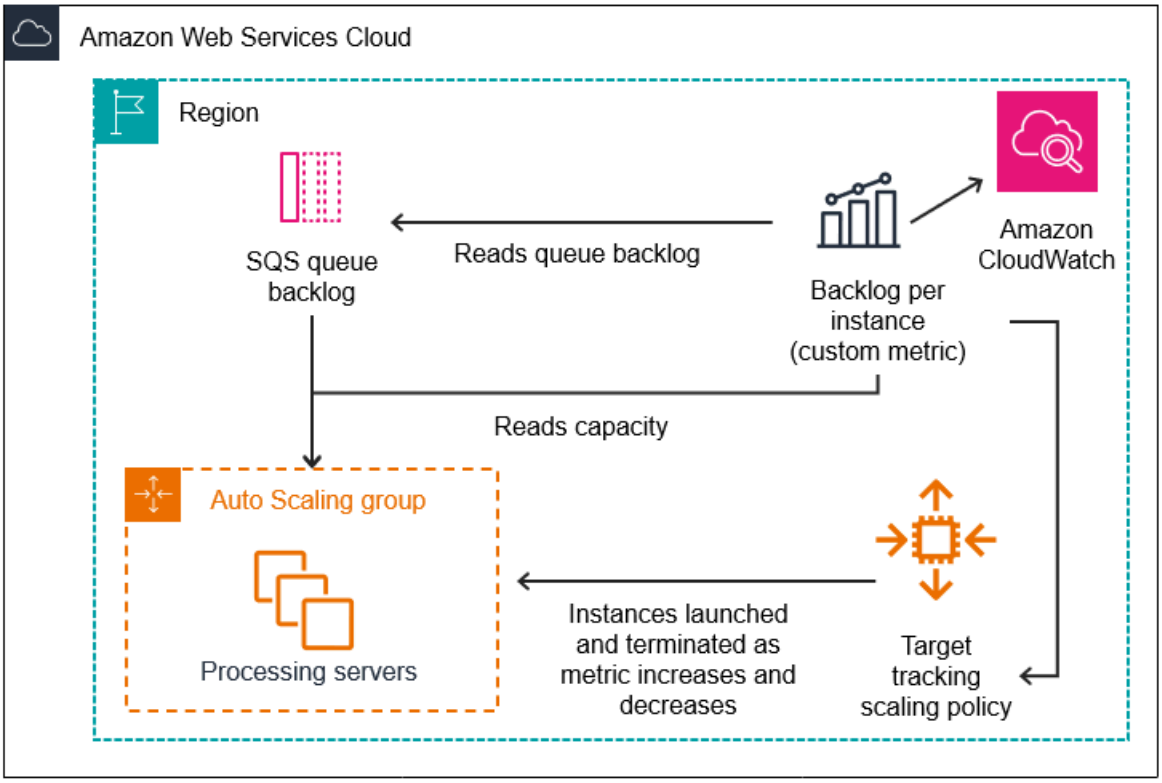
En los procedimientos siguientes se muestra cómo publicar la métrica personalizada y crear la política de escalado de seguimiento de destino que configura el grupo de escalado automático para que el escalado se realice en función de estos cálculos.

> **Important**
>
> Recuerde que, para reducir los costos, puede utilizar la calculadora de métricas. Para obtener más información, consulte Creación de una política de escalado de seguimiento de destino con cálculos métricos.

Existen tres puntos principales para esta configuración:

- Un grupo de escalado automático para administrar las instancias EC2 para procesar mensajes de una cola de SQS.
- Una métrica personalizada para enviar a Amazon CloudWatch que mide el número de mensajes en la cola por instancia EC2 del grupo Auto Scaling.
- Una política de seguimiento de objetivos que configura su grupo de Auto Scaling para que escale en función de la métrica personalizada y un valor objetivo establecido. CloudWatch las alarmas invocan la política de escalado.

El siguiente diagrama ilustra la arquitectura de esta configuración.



Limitaciones

Debe usar el AWS CLI o un SDK para publicar su métrica personalizada. CloudWatch A continuación, puede supervisar su métrica con el Consola de administración de AWS.

En las siguientes secciones, utiliza el AWS CLI para las tareas que necesita realizar. Por ejemplo, para obtener datos métricos que reflejen el uso actual de la cola, utilice el comando SQS. get-queue-attributes

Configuración del escalado basada en Amazon SQS

El siguiente procedimiento describe la forma de configurar el escalado automático basado en Amazon SQS. Aprenderá a crear una métrica CloudWatch personalizada, a configurar una política de seguimiento de objetivos mediante ella y a probar su configuración. AWS CLI

Antes de empezar, asegúrate de que la tienes AWS CLI instalada y configurada. Además, debe tener una cola de Amazon SQS que pueda usar. En las siguientes tareas se presupone que ya tiene una cola (estándar o FIFO), un grupo de escalado automático e instancias EC2 ejecutándose en la aplicación que utiliza la cola.

Para obtener más información sobre Amazon SQS, consulte la Guía del desarrollador de Amazon Simple Queue Service.

Tareas

- Paso 1: Crea una métrica CloudWatch personalizada
- Paso 2: Crear una política de escalado de seguimiento de destino
- Paso 3: Prueba de la política de escalado

Paso 1: Crea una métrica CloudWatch personalizada

Una métrica personalizada se define mediante un nombre de métrica y un espacio de nombres de su elección. Los espacios de nombres para métricas personalizadas no pueden comenzar por `AWS/`. Para obtener más información sobre la publicación de métricas personalizadas, consulta el tema Publicar métricas personalizadas en la Guía del CloudWatch usuario de Amazon.

Siga este procedimiento para crear la métrica personalizada leyendo primero la información de su AWS cuenta. A continuación, calcule las tareas pendientes para cada métrica de instancia, tal como se recomienda en una sección anterior. Por último, publique este número con CloudWatch una precisión de 1 minuto. Siempre que sea posible, le recomendamos que cuando realizado el escalado basado en métricas, utilice una granularidad de un minuto para garantizar una respuesta más rápida a los cambios en la carga del sistema.

Para crear una métrica CloudWatch personalizada ()AWS CLI

1. Utilice el comando get-queue-attributes de SQS para obtener el número de mensajes en espera en la cola (`ApproximateNumberOfMessages`).

   ```
   aws sqs get-queue-attributes --queue-url https://
   sqs.region.amazonaws.com/123456789/MyQueue \
     --attribute-names ApproximateNumberOfMessages
   ```

2. Utilice el comando describe-auto-scaling-groups para obtener la capacidad de ejecución del grupo, que es el número de instancias que tienen el estado de ciclo de vida `InService`. Este comando devuelve las instancias de un grupo de escalado automático junto con su estado de ciclo de vida.

   ```
   aws autoscaling describe-auto-scaling-groups --auto-scaling-group-names my-asg
   ```

3. Calcule las tareas pendientes por instancia dividiendo el número aproximado de mensajes disponibles para su recuperación de la cola por la capacidad de puesta en marcha del grupo.
4. Cree un script que se ejecute cada minuto para recuperar el valor acumulado por instancia y publicarlo en una métrica CloudWatch personalizada. Cuando publica una métrica personalizada, especifica el nombre de la métrica, el espacio de nombres, la unidad, el valor y cero o más dimensiones. Una dimensión consta de un nombre de dimensión y un valor de dimensión.

   Para publicar tu métrica personalizada, sustituye los valores de los marcadores de posición por el nombre de la métrica que prefieras, el valor de la métrica, un espacio de nombres (siempre que no empiece por AWS «) y las dimensiones (opcional) y, a continuación, ejecuta el siguiente comando. *italics* put-metric-data

   ```
   aws cloudwatch put-metric-data --metric-name MyBacklogPerInstance --
   namespace MyNamespace \
     --unit None --value 20 --
   dimensions MyOptionalMetricDimensionName=MyOptionalMetricDimensionValue
   ```

Después de que la aplicación emita la métrica deseada, los datos se envían a CloudWatch. La métrica está visible en la consola. CloudWatch Puede acceder a ella iniciando sesión en la CloudWatch página Consola de administración de AWS y navegando hasta ella. A continuación, puede ver la métrica desplazándose a la página de métricas o buscándola usando el campo de búsqueda. Para obtener información sobre la visualización de las métricas, consulta Ver las métricas disponibles en la Guía del CloudWatch usuario de Amazon.

Paso 2: Crear una política de escalado de seguimiento de destino

La métrica que creó ahora se puede añadir a una política de escalado de seguimiento de destino.

Creación de una política de escalado de seguimiento de destino (AWS CLI)

1. Utilice el siguiente comando `cat` para almacenar un valor de destino para su política de escalado y una especificación de métricas personalizada en un archivo JSON llamado `config.json` en su directorio principal. Reemplace cada *user input placeholder* por su propia información. Para el `TargetValue`, calcule las tareas pendientes aceptables por cada métrica de instancia e introdúzcala aquí. Para calcular este número, decida un valor de latencia normal y divídalo por el tiempo medio que tarda en procesar un mensaje, como se describe en una sección anterior.

Si no especificó ninguna dimensión para la métrica que creó en el paso 1, no incluya ninguna dimensión en la especificación métrica personalizada.

```
$ cat ~/config.json
{
   "TargetValue":100,
   "CustomizedMetricSpecification":{
      "MetricName":"MyBacklogPerInstance",
      "Namespace":"MyNamespace",
      "Dimensions":[
         {
            "Name":"MyOptionalMetricDimensionName",
            "Value":"MyOptionalMetricDimensionValue"
         }
      ],
      "Statistic":"Average",
      "Unit":"None"
   }
}
```

2. Utilice el comando put-scaling-policy, junto con el archivo `config.json` creado en el paso anterior, para crear la política de escalado.

```
aws autoscaling put-scaling-policy --policy-name sqs100-target-tracking-scaling-
policy \
  --auto-scaling-group-name my-asg --policy-type TargetTrackingScaling \
  --target-tracking-configuration file://~/config.json
```

Esto crea dos alarmas: una para el escalado horizontal y otra para la reducción horizontal. También devuelve el nombre de recurso de Amazon (ARN) de la política en la que está registrada CloudWatch, que se CloudWatch utiliza para invocar el escalado cada vez que se infringe el umbral métrico.

Paso 3: Prueba de la política de escalado

Una vez que finalice la configuración, verifique que la política de escalado funcione. Puede probarla aumentando el número de mensajes en la cola de SQS y verificando después que el grupo de escalado automático ha lanzado una instancia EC2 adicional. También puede probarla reduciendo el

número de mensajes en la cola de SQS y verificando después que el grupo de escalado automático ha terminado una instancia EC2.

Pruebas de la función de escalado horizontal

1. Siga los pasos de Creación de una cola estándar de Amazon SQS y envío de un mensaje o Creación de una cola FIFO de Amazon SQS y envío de un mensaje para añadir mensajes a la cola. Asegúrese de que ha aumentado el número de mensajes en la cola de forma que las tareas pendientes por cada métrica de instancia supere el valor de destino.

   Este proceso puede tardar unos minutos hasta que los cambios invoquen la alarma.
2. Utilice el comando describe-auto-scaling-groups para comprobar que el grupo ha lanzado una instancia.

   ```
   aws autoscaling describe-auto-scaling-groups --auto-scaling-group-name my-asg
   ```

Cómo probar la función de reducción horizontal

1. Siga los pasos de Recibir y eliminar un mensaje (consola) para eliminar los mensajes de la cola. Asegúrese de que ha disminuido el número de mensajes en la cola de forma que las tareas pendientes por cada métrica de instancia esté por debajo del valor de destino.

   Este proceso puede tardar unos minutos hasta que los cambios invoquen la alarma.
2. Utilice el comando describe-auto-scaling-groups para comprobar que el grupo ha terminado una instancia.

   ```
   aws autoscaling describe-auto-scaling-groups --auto-scaling-group-name my-asg
   ```

Amazon SQS y protección de la reducción horizontal

Los mensajes que no se procesaron cuando se terminó la instancia se devuelven a la cola de SQS, donde pueden ser procesados por otra instancia que siga ejecutándose. Para las aplicaciones en las que se realizan tareas de ejecución prolongada, puede utilizar opcionalmente la protección frente a la reducción horizontal de instancias para tener el control sobre los procesos de trabajo de la cola que se terminan cuando el grupo de escalado automático se reduce horizontalmente.

En el siguiente pseudocódigo se muestra una forma de proteger los procesos de trabajo controlados por cola de larga ejecución frente a la terminación por reducción horizontal.

```
while (true)
{
  SetInstanceProtection(False);
  Work = GetNextWorkUnit();
  SetInstanceProtection(True);
  ProcessWorkUnit(Work);
  SetInstanceProtection(False);
}
```

Para obtener más información, consulte Diseño de aplicaciones para gestionar sin problemas la terminación de instancias.

Verificación de una actividad de escalado para un grupo de escalado automático

En la sección Amazon EC2 Auto Scaling de la consola de Amazon EC2, el Activity history (Historial de actividades) de un grupo de escalado automático permite ver el estado actual de una actividad de escalado que se encuentre en curso. Una vez finalizada la actividad de escalado, podrá comprobar si se ha realizado o no correctamente. Esta opción es especialmente práctica cuando se crean grupos de Auto Scaling o se agregan condiciones de escalado a grupos existentes.

Cuando agrega una política de seguimiento de destino, de pasos o de escalado simple a su grupo de escalado automático, Amazon EC2 Auto Scaling comienza a evaluar de inmediato la política en función de la métrica. La alarma de la métrica pasa al estado ALARM cuando la métrica supera el umbral durante un número especificado de periodos de evaluación. Esto significa que una política de escalado podría dar lugar a una actividad de escalado poco después de crearla. Después de que Amazon EC2 Auto Scaling ajuste la capacidad deseada en respuesta a una política de escalado, puede verificar la actividad de escalado en su cuenta. Si desea recibir una notificación por email de Amazon EC2 Auto Scaling que le informe sobre una actividad de escalado de escalado, siga las instrucciones de Opciones de notificaciones de Amazon SNS para Amazon EC2 Auto Scaling.

> **Tip**
>
> En el siguiente procedimiento, observará las secciones Activity history (Historial de actividad) e Instances (Instancias) del grupo de escalado automático. En ambas, ya deberían aparecer las columnas con nombre. Para mostrar las columnas ocultas o cambiar el número de

filas que aparecen, elija el icono de engranaje en la esquina superior derecha de cada sección para abrir el modal de preferencias, actualice la configuración según sea necesario y seleccione Confirm (Confirmar).

Para ver las actividades de escalado de un grupo de escalado automático (consola)

1. Abra la consola Amazon EC2 en https://console.aws.amazon.com/ec2/y seleccione Auto Scaling Groups en el panel de navegación.
2. En la barra de navegación de la parte superior de la pantalla, elija la Región en la que se encuentra su grupo de escalado automático.
3. Seleccione la casilla situada junto al grupo de escalado automático.

   Se abre un panel dividido en la parte inferior de la página.
4. En la pestaña Activity (Actividad), en Activity history (Historial de actividad), la columna Status (Estado) muestra si su grupo de escalado automático ha lanzado las instancias o las ha terminado correctamente, o bien si la actividad de escalado sigue en curso.
5. (Opcional) Si tiene muchas actividades de escalado, puede elegir el icono > en el borde superior del historial de actividades y así acceder a la siguiente página de actividades de escalado.
6. En la pestaña Instance management (Administración de instancias), en Instances (Instancias), la columna Lifecycle (Ciclo de vida) muestra el estado de sus instancias. Una vez que se inicia la instancia y los enlaces de ciclo de vida han finalizado, su estado de ciclo de vida cambia a `InService`. La columna Health status (Estado) muestra el resultado de la comprobación de estado de instancias EC2 correspondiente a su instancia.

Para ver las actividades de escalado de un grupo de escalado automático (AWS CLI)

Use el siguiente comando describe-scaling-activities.

```
aws autoscaling describe-scaling-activities --auto-scaling-group-name my-asg
```

A continuación, se muestra un ejemplo del resultado.

Las actividades de escalado se ordenan por hora de inicio. En primer lugar, se describen las actividades aún en curso.

```
{
  "Activities": [
```

```
    {
      "ActivityId": "5e3a1f47-2309-415c-bfd8-35aa06300799",
      "AutoScalingGroupName": "my-asg",
      "Description": "Terminating EC2 instance: i-06c4794c2499af1df",
      "Cause": "At 2020-02-11T18:34:10Z a monitor alarm TargetTracking-my-asg-AlarmLow-
b9376cab-18a7-4385-920c-dfa3f7783f82 in state ALARM triggered policy my-target-
tracking-policy changing the desired capacity from 3 to 2.  At 2020-02-11T18:34:31Z
 an instance was taken out of service in response to a difference between desired and
 actual capacity, shrinking the capacity from 3 to 2.  At 2020-02-11T18:34:31Z instance
 i-06c4794c2499af1df was selected for termination.",
      "StartTime": "2020-02-11T18:34:31.268Z",
      "EndTime": "2020-02-11T18:34:53Z",
      "StatusCode": "Successful",
      "Progress": 100,
      "Details": "{\"Subnet ID\":\"subnet-5ea0c127\",\"Availability Zone\":\"us-west-2a
\"...}",
      "AutoScalingGroupARN": "arn"
    },
...
  ]
}
```

Para obtener una descripción de los campos de la salida, consulte Actividad en la Referencia de API de Amazon EC2 Auto Scaling.

Para obtener ayuda para recuperar las actividades de escalado de un grupo eliminado y obtener información acerca de los tipos de errores que puede encontrar y cómo gestionarlos, consulte Solución de problemas de Amazon EC2 Auto Scaling.

Desactivación de una política de escalado para un grupo de escalado automático

En este tema se describe cómo desactivar temporalmente una política de escalado para que no inicie cambios en el número de instancias que contiene el grupo de escalado automático. Cuando deshabilita una política de escalado, los detalles de configuración se conservan, de modo que puede volver a habilitar rápidamente la política. Esto es más fácil que eliminar temporalmente una política cuando no la necesita y volver a crearla más tarde.

Cuando se desactiva una política de escalado, el grupo de escalado automático no escala ni reduce horizontalmente en las alarmas de métrica que se interrumpen mientras la política de escalado está desactivada. Sin embargo, las actividades de escalado que siguen en curso no se detienen.

Tenga en cuenta que las políticas de escalado desactivadas se siguen considerando para las cuotas del número de políticas de escalado que puede agregar a un grupo de escalado automático.

Para desactivar una política de escalado (consola)

1. Abra la consola Amazon EC2 en https://console.aws.amazon.com/ec2/y seleccione Auto Scaling Groups en el panel de navegación.
2. Seleccione la casilla situada junto al grupo de escalado automático.

   Se abre un panel dividido en la parte inferior de la página.
3. En la pestaña Automatic scaling (Escalado automático), en Dynamic scaling policies (Políticas de escalado dinámico), seleccione la casilla situada en la esquina superior derecha de la política de escalado deseada.
4. Desplácese hasta la parte superior de la sección Dynamic scaling policies (Políticas de escalado dinámico) y, a continuación, elija Actions (Acciones), Disable (Desactivar).

Cuando esté listo para volver a habilitar la política de escalado, repita estos pasos y, a continuación, elija Actions (Acciones), Enable (Habilitar). Después de volver a habilitar una política de escalado, el grupo de escalado automático puede iniciar inmediatamente una acción de escalado si hay alguna alarma actualmente en estado ALARMA.

Para deshabilitar una política de escalado (AWS CLI)

Utilice el comando put-scaling-policy con la opción `--no-enabled`, como se indica a continuación. Especifique todas las opciones en el comando tal como las especificaría al crear la política.

```
aws autoscaling put-scaling-policy --auto-scaling-group-name my-asg \
   --policy-name my-scaling-policy --policy-type TargetTrackingScaling \
   --estimated-instance-warmup 360 \
   --target-tracking-configuration '{ "TargetValue": 70,
 "PredefinedMetricSpecification": { "PredefinedMetricType":
 "ASGAverageCPUUtilization" } }' \
   --no-enabled
```

Para volver a habilitar una política de escalado (AWS CLI)

Utilice el comando put-scaling-policy con la opción `--enabled`, como se indica a continuación. Especifique todas las opciones en el comando tal como las especificaría al crear la política.

```
aws autoscaling put-scaling-policy --auto-scaling-group-name my-asg \
   --policy-name my-scaling-policy --policy-type TargetTrackingScaling \
   --estimated-instance-warmup 360 \
   --target-tracking-configuration '{ "TargetValue": 70,
 "PredefinedMetricSpecification": { "PredefinedMetricType":
 "ASGAverageCPUUtilization" } }' \
   --enabled
```

Para describir una política de escalado (AWS CLI)

Utilice el comando describe-policies para verificar el estado habilitado de una política de escalado.

```
aws autoscaling describe-policies --auto-scaling-group-name my-asg \
   --policy-names my-scaling-policy
```

A continuación, se muestra un ejemplo del resultado.

```
{
    "ScalingPolicies": [
        {
            "AutoScalingGroupName": "my-asg",
            "PolicyName": "my-scaling-policy",
            "PolicyARN": "arn:aws:autoscaling:us-
west-2:123456789012:scalingPolicy:1d52783a-b03b-4710-
bb0e-549fd64378cc:autoScalingGroupName/my-asg:policyName/my-scaling-policy",
            "PolicyType": "TargetTrackingScaling",
            "StepAdjustments": [],
            "Alarms": [
                {
                    "AlarmName": "TargetTracking-my-asg-
AlarmHigh-9ca53fdd-7cf5-4223-938a-ae1199204502",
                    "AlarmARN": "arn:aws:cloudwatch:us-
west-2:123456789012:alarm:TargetTracking-my-asg-AlarmHigh-9ca53fdd-7cf5-4223-938a-
ae1199204502"
                },
                {
                    "AlarmName": "TargetTracking-my-asg-AlarmLow-7010c83d-d55a-4a7a-
abe0-1cf8b9de6d6c",
                    "AlarmARN": "arn:aws:cloudwatch:us-
west-2:123456789012:alarm:TargetTracking-my-asg-AlarmLow-7010c83d-d55a-4a7a-
abe0-1cf8b9de6d6c"
                }
            ],
```

```
            "TargetTrackingConfiguration": {
                "PredefinedMetricSpecification": {
                    "PredefinedMetricType": "ASGAverageCPUUtilization"
                },
                "TargetValue": 70.0,
                "DisableScaleIn": false
            },
            "Enabled": true
        }
    ]
}
```

Eliminación de una política de escalado para un grupo de escalado automático

Puede eliminar una política de escalado cuando ya no la necesite. Según el tipo de política de escalado, es posible que también necesite eliminar las CloudWatch alarmas. Al eliminar una política de escalado y seguimiento de objetivos, también se eliminan todas CloudWatch las alarmas asociadas. Al eliminar una política de escalado escalonado o una política de escalado simple, se elimina la acción de alarma subyacente, pero no se elimina la CloudWatch alarma, incluso si ya no tiene una acción asociada.

Para eliminar una política de escalado (consola)

1. Abra la consola Amazon EC2 en https://console.aws.amazon.com/ec2/y seleccione Auto Scaling Groups en el panel de navegación.
2. Seleccione la casilla situada junto al grupo de escalado automático.

   Se abre un panel dividido en la parte inferior de la página.
3. En la pestaña Automatic scaling (Escalado automático), en Dynamic scaling policies (Políticas de escalado dinámico), seleccione la casilla situada en la esquina superior derecha de la política de escalado deseada.
4. Desplácese hasta la parte superior de la sección Dynamic scaling policies (Políticas de escalado dinámico) y, a continuación, elija Actions (Acciones), Delete (Eliminar).
5. Cuando se le indique que confirme, seleccione Yes, Delete (Sí, borrar).
6. (Opcional) Si ha eliminado una política de escalado escalonado o una política de escalado simple, haga lo siguiente para eliminar la CloudWatch alarma asociada a la política. Puede omitir estos pasos secundarios para mantener la alarma para uso futuro.

a. Abra la CloudWatch consola en https://console.aws.amazon.com/cloudwatch/.
b. En el panel de navegación, elija Alarms.
c. Elija la alarma (por ejemplo, `Step-Scaling-AlarmHigh-AddCapacity`) y elija Action (Acción), Delete (Eliminar).
d. Cuando se le pida confirmación, seleccione Eliminar.

Para obtener las políticas de escalado de un grupo de escalado automático (AWS CLI)

Antes de eliminar una política de escalado, utilice el siguiente comando describe-policies para ver qué políticas de escalado se crearon para el grupo de escalado automático. Puede utilizar la salida al eliminar la política y las alarmas de CloudWatch.

```
aws autoscaling describe-policies --auto-scaling-group-name my-asg
```

Puede filtrar los resultados por el tipo de política de escalado mediante el parámetro `--query`. Esta sintaxis de `query` funciona en Linux y macOS. En Windows, cambie la comillas simples por comillas dobles.

```
aws autoscaling describe-policies --auto-scaling-group-name my-asg
  --query 'ScalingPolicies[?PolicyType==`TargetTrackingScaling`]'
```

A continuación, se muestra un ejemplo del resultado.

```
[
    {
        "AutoScalingGroupName": "my-asg",
        "PolicyName": "cpu50-target-tracking-scaling-policy",
        "PolicyARN": "PolicyARN",
        "PolicyType": "TargetTrackingScaling",
        "StepAdjustments": [],
        "Alarms": [
            {
                "AlarmARN": "arn:aws:cloudwatch:us-
west-2:123456789012:alarm:TargetTracking-my-asg-AlarmHigh-
fc0e4183-23ac-497e-9992-691c9980c38e",
                "AlarmName": "TargetTracking-my-asg-AlarmHigh-
fc0e4183-23ac-497e-9992-691c9980c38e"
            },
```

```
            {
                "AlarmARN": "arn:aws:cloudwatch:us-
west-2:123456789012:alarm:TargetTracking-my-asg-AlarmLow-61a39305-ed0c-47af-
bd9e-471a352ee1a2",
                "AlarmName": "TargetTracking-my-asg-AlarmLow-61a39305-ed0c-47af-
bd9e-471a352ee1a2"
            }
        ],
        "TargetTrackingConfiguration": {
            "PredefinedMetricSpecification": {
                "PredefinedMetricType": "ASGAverageCPUUtilization"
            },
            "TargetValue": 50.0,
            "DisableScaleIn": false
        },
        "Enabled": true
    }
]
```

Para eliminar una política de escalado (AWS CLI)

Use el siguiente comando delete-policy.

```
aws autoscaling delete-policy --auto-scaling-group-name my-asg \
  --policy-name cpu50-target-tracking-scaling-policy
```

Para eliminar la CloudWatch alarma (AWS CLI)

Para políticas de escalado sencillas y escalonadas, utilice el comando delete-alarm para eliminar la CloudWatch alarma asociada a la política. Puede omitir este paso si desea mantener la alarma para usarla en el futuro. Puede eliminar una o más alarmas a la vez. Por ejemplo, utilice el siguiente comando para eliminar las alarmas `Step-Scaling-AlarmHigh-AddCapacity` y `Step-Scaling-AlarmLow-RemoveCapacity`.

```
aws cloudwatch delete-alarms --alarm-name Step-Scaling-AlarmHigh-AddCapacity Step-
Scaling-AlarmLow-RemoveCapacity
```

Ejemplos de políticas de escalado para AWS CLI

Puede crear políticas de escalado para Amazon EC2 Auto Scaling mediante Consola de administración de AWS AWS Command Line Interface ,AWS CLI() o. SDKs

Los siguientes ejemplos muestran cómo puede crear políticas de escalado para Amazon EC2 Auto Scaling con AWS CLI put-scaling-policyel comando. Reemplace cada *user input placeholder* por su propia información.

Para empezar a escribir políticas de escalado mediante el AWS CLI, consulte los ejercicios introductorios en Políticas de escalado de seguimiento de destino yPolíticas de escalado sencillo y por pasos.

Ejemplo 1: aplicar una política de escalado de seguimiento de destino con una especificación de métrica predefinida

```
aws autoscaling put-scaling-policy --policy-name cpu50-target-tracking-scaling-policy \
  --auto-scaling-group-name my-asg --policy-type TargetTrackingScaling \
  --target-tracking-configuration file://config.json
{
  "TargetValue": 50.0,
  "PredefinedMetricSpecification": {
    "PredefinedMetricType": "ASGAverageCPUUtilization"
  }
}
```

Para obtener más información, consulte la PredefinedMetricSpecificationreferencia de la API Amazon EC2 Auto Scaling.

> **Note**
>
> Si el archivo no se encuentra en el directorio actual, escriba la ruta completa al archivo. Para obtener más información sobre la lectura de los valores de los AWS CLI parámetros de un archivo, consulte Cargar AWS CLI parámetros desde un archivo en la Guía del AWS Command Line Interface usuario.

Ejemplo 2: aplicar una política de escalado de seguimiento de destino con una especificación de métrica personalizada

```
aws autoscaling put-scaling-policy --policy-name sqs100-target-tracking-scaling-policy
 \
  --auto-scaling-group-name my-asg --policy-type TargetTrackingScaling \
  --target-tracking-configuration file://config.json
{
```

```
  "TargetValue": 100.0,
  "CustomizedMetricSpecification": {
    "MetricName": "MyBacklogPerInstance",
    "Namespace": "MyNamespace",
    "Dimensions": [{
      "Name": "MyOptionalMetricDimensionName",
      "Value": "MyOptionalMetricDimensionValue"
    }],
    "Statistic": "Average",
    "Unit": "None"
  }
}
```

Para obtener más información, consulte la CustomizedMetricSpecificationreferencia de la API Amazon EC2 Auto Scaling.

Ejemplo 3: aplicar una política de escalado de seguimiento de destino solo para el escalado ascendente

```
aws autoscaling put-scaling-policy --policy-name alb1000-target-tracking-scaling-policy
 \
  --auto-scaling-group-name my-asg --policy-type TargetTrackingScaling \
  --target-tracking-configuration file://config.json
{
  "TargetValue": 1000.0,
  "PredefinedMetricSpecification": {
    "PredefinedMetricType": "ALBRequestCountPerTarget",
    "ResourceLabel": "app/my-alb/778d41231b141a0f/targetgroup/my-alb-target-
group/943f017f100becff"
  },
  "DisableScaleIn": true
}
```

Ejemplo 4: aplicar una política de escalado por pasos para el escalado ascendente

```
aws autoscaling put-scaling-policy \
  --auto-scaling-group-name my-asg  \
  --policy-name my-step-scale-out-policy \
  --policy-type StepScaling \
  --adjustment-type PercentChangeInCapacity \
  --metric-aggregation-type Average \
  --step-adjustments
 MetricIntervalLowerBound=10.0,MetricIntervalUpperBound=20.0,ScalingAdjustment=10 \
```

```
 MetricIntervalLowerBound=20.0,MetricIntervalUpperBound=30.0,ScalingAdjustment=20 \
                      MetricIntervalLowerBound=30.0,ScalingAdjustment=30 \
  --min-adjustment-magnitude 1
```

Registre el nombre de recurso de Amazon (ARN) de la política. Necesitará el ARN al crear la CloudWatch alarma.

Ejemplo 5: aplicar una política de escalado por pasos para la reducción horizontal

```
aws autoscaling put-scaling-policy \
  --auto-scaling-group-name my-asg  \
  --policy-name my-step-scale-in-policy \
  --policy-type StepScaling \
  --adjustment-type ChangeInCapacity \
  --step-adjustments MetricIntervalUpperBound=0.0,ScalingAdjustment=-2
```

Registre el nombre de recurso de Amazon (ARN) de la política. Necesitará el ARN al crear la CloudWatch alarma.

Ejemplo 6: aplicar una política de escalado sencillo para el escalado ascendente

```
aws autoscaling put-scaling-policy --policy-name my-simple-scale-out-policy \
  --auto-scaling-group-name my-asg --scaling-adjustment 30 \
  --adjustment-type PercentChangeInCapacity --min-adjustment-magnitude 2
```

Registre el nombre de recurso de Amazon (ARN) de la política. Necesitará el ARN al crear la CloudWatch alarma.

Ejemplo 7: aplicar una política de escalado sencillo para la reducción horizontal

```
aws autoscaling put-scaling-policy --policy-name my-simple-scale-in-policy \
  --auto-scaling-group-name my-asg --scaling-adjustment -1 \
  --adjustment-type ChangeInCapacity --cooldown 180
```

Registre el nombre de recurso de Amazon (ARN) de la política. Necesitará el ARN al crear la CloudWatch alarma.

Escalado predictivo para Amazon EC2 Auto Scaling

El escalado predictivo funciona mediante el análisis de los datos históricos de carga para detectar patrones diarios o semanales en los flujos de tráfico. Utiliza esta información para pronosticar las necesidades de capacidad futuras, de modo que Amazon EC2 Auto Scaling pueda aumentar proactivamente la capacidad de su grupo de escalado automático para adaptarla a la carga prevista.

El escalado predictivo es adecuado para situaciones en las que tiene:

- Tráfico cíclico; por ejemplo, un uso elevado de recursos durante el horario laborable normal y un uso reducido de recursos por la noche o los fines de semana.
- Patrones recurrentes de carga de trabajo de activación y desactivación, como procesamiento por lotes, pruebas o análisis periódicos de datos.
- Aplicaciones que tardan mucho tiempo en inicializarse, lo que provoca un notable impacto de latencia en el rendimiento de las aplicaciones durante eventos de escalado horizontal.

En general, si tiene patrones regulares de aumento de tráfico y aplicaciones que tardan mucho tiempo en inicializarse, debe considerar el uso del escalado predictivo. El escalado predictivo puede ayudarle a escalar más rápidamente al lanzar la capacidad antes de la carga prevista, en comparación con solo usar el escalado dinámico, que tiene una naturaleza reactiva. Además, el escalado predictivo puede ahorrarle dinero en la factura de EC2, ya que le ayuda a evitar la necesidad de aprovisionar la capacidad en exceso.

Por ejemplo, piense en una aplicación que tiene un uso elevado durante el horario laborable y un uso bajo durante la noche. Al comienzo de cada día laborable, el escalado predictivo puede agregar capacidad antes de la primera afluencia de tráfico. Esto ayuda a la aplicación a mantener una alta disponibilidad y rendimiento al pasar de un periodo de menor utilización a uno de mayor utilización. No tiene que esperar a que el escalado dinámico reaccione a los cambios en el tráfico. Tampoco tiene que dedicar tiempo a revisar los patrones de carga de la aplicación e intentar programar la cantidad correcta de capacidad mediante el escalado programado.

Temas

- Funcionamiento del escalado predictivo
- Creación de una política de escalado predictivo para un grupo de escalado automático
- Evaluación de las políticas de escalado predictivo
- Anulación de valores de pronóstico mediante acciones programadas

- Políticas avanzadas de escalado predictivo mediante métricas personalizadas

Funcionamiento del escalado predictivo

Este tema explica cómo funciona el escalado predictivo y describe qué debe tener en cuenta al crear una política de escalado predictivo.

Temas

- Funcionamiento
- Límite de la capacidad máxima
- Consideraciones
- Regiones admitidas

Funcionamiento

Para utilizar el escalado predictivo, cree una política de escalado predictivo que especifique la CloudWatch métrica que se va a supervisar y analizar. Para que el escalado predictivo comience a pronosticar valores futuros, esta métrica debe tener datos de, por lo menos, las últimas 24 horas.

Tras crear la política, el escalado predictivo comienza a analizar los datos de las métricas de los últimos 14 días para identificar patrones. Utiliza este análisis para generar una previsión horaria de los requisitos de capacidad para las próximas 48 horas. La previsión se actualiza cada 6 horas con los CloudWatch datos más recientes. A medida que llegan nuevos datos, el escalado predictivo puede mejorar continuamente la precisión de las previsiones futuras.

La primera vez que se habilita el escalado predictivo, se ejecuta en modo de solo previsión. En este modo, genera previsiones de capacidad, pero no escala el grupo de escalado automático en función de esas previsiones. Esto permite evaluar la precisión e idoneidad de la previsión. Puede ver los datos del pronóstico mediante la operación de API `GetPredictiveScalingForecast` o la Consola de administración de AWS.

Tras revisar los datos de previsión y decidir empezar a escalar en función de esos datos, cambie la política de escalado al modo de previsión y escalado. En este modo:

- Si la previsión prevé un aumento de la carga, Amazon EC2 Auto Scaling aumentará la capacidad mediante el escalado horizontal.

- Si la previsión prevé una disminución de la carga, no se reducirá la capacidad mediante la reducción horizontal. Si desea eliminar la capacidad que ya no necesita, debe crear políticas de escalado dinámico.

De forma predeterminada, Amazon EC2 Auto Scaling escala el grupo de escalado automático al principio de cada hora en función de la previsión de esa hora. Si lo desea, puede especificar una hora de inicio anterior mediante la `SchedulingBufferTime` propiedad de la operación de la `PutScalingPolicy` API o la configuración de Pre-launch instancias de Consola de administración de AWS. Esto hace que Amazon EC2 Auto Scaling lance nuevas instancias anticipándose a la demanda prevista, lo que les da tiempo para arrancar y prepararse para gestionar el tráfico.

Para permitir el lanzamiento de instancias nuevas antes de la demanda prevista, se recomienda encarecidamente habilitar la preparación predeterminada de instancias para su grupo de escalado automático. Esto especifica un período de tiempo posterior a una actividad de escalado horizontal durante el cual Amazon EC2 Auto Scaling no realizará una reducción horizontal, incluso si las políticas de escalado dinámico indican que la capacidad debe reducirse. Esto ayuda a garantizar que las instancias recién lanzadas dispongan del tiempo suficiente para empezar a atender el aumento del tráfico antes de que se las considere para operaciones de reducción horizontal. Para obtener más información, consulte Establecimiento de la preparación predeterminada de instancias para un grupo de escalado automático.

Límite de la capacidad máxima

Los grupos de escalado automático tienen una configuración de capacidad máxima que limita el número máximo de instancias de EC2 que se pueden lanzar para el grupo. De manera predeterminada, cuando se establecen políticas de escalado, estas no pueden aumentar la capacidad por encima de la capacidad máxima.

Como alternativa, puede permitir que la capacidad máxima del grupo aumente automáticamente si la capacidad previste se aproxima o supera la capacidad máxima del grupo de escalado automático. Para habilitar este comportamiento, utilice las propiedades `MaxCapacityBreachBehavior` y `MaxCapacityBuffer` de la operación de API `PutScalingPolicy` o la configuración de Comportamiento de capacidad máxima de la Consola de administración de AWS.

> **Warning**
>
> Tenga cuidado al permitir que la capacidad máxima aumente automáticamente. Esto puede provocar el lanzamiento de más instancias de las previstas si no se supervisa ni gestiona

el aumento de la capacidad máxima. La capacidad máxima incrementada se convertirá en la nueva capacidad máxima normal para el grupo de escalado automático hasta que la actualice manualmente. La capacidad máxima no vuelve a disminuir automáticamente hasta el máximo original.

Consideraciones

- Confirme si el escalado predictivo es adecuado para su carga de trabajo. Una carga de trabajo es idónea para el escalado predictivo si presenta patrones de carga recurrentes específicos del día de la semana o de la hora del día. Para comprobar esto, configure las políticas de escalado predictivo en el modo solo previsión y, a continuación, consulte las recomendaciones de la consola. Amazon EC2 Auto Scaling ofrece recomendaciones basadas en observaciones sobre el posible rendimiento de las políticas. Evalúe la previsión y su precisión antes de permitir que el escalado predictivo escale la aplicación de forma activa.
- Para comenzar el pronóstico, el escalado predictivo necesita al menos 24 horas de datos históricos. Sin embargo, las previsiones son más eficaces si los datos históricos abarcan dos semanas completas. Si para actualizar la aplicación crea un nuevo grupo de escalado automático y elimina el anterior, el nuevo grupo de escalado automático necesita 24 horas de datos de carga históricos antes de que el escalado predictivo pueda volver a generar pronósticos. Puede usar métricas personalizadas para agregar métricas de grupos de escalado automático nuevos y antiguos. De lo contrario, es posible que tenga que esperar unos días para obtener una previsión más precisa.
- Elija una métrica de carga que represente con precisión la carga total de la aplicación y que sea el aspecto de la aplicación que más le interese escalar.
- El uso del escalado dinámico en conjunto con el escalado predictivo le ayuda a seguir de cerca la curva de demanda de su aplicación, reduciendo horizontalmente durante periodos de bajo tráfico y escalando horizontalmente cuando el tráfico es mayor de lo esperado. Cuando hay activas varias políticas de escalado, cada política determina la capacidad deseada de forma independiente, y la capacidad deseada se establece en el máximo de ellas. Por ejemplo, si se requieren 10 instancias para mantenerse en la utilización de destino en una política de escalado de seguimiento de destino y se requieren 8 instancias para mantenerse en la utilización de destino en una política de escalado predictivo, la capacidad deseada del grupo se establece en 10. Si no está familiarizado con el escalado dinámico, le recomendamos que utilice políticas de escalado de seguimiento de objetivo. Para obtener más información, consulte Escalado dinámico para Amazon EC2 Auto Scaling.

- Una suposición fundamental del escalado predictivo es que el grupo de escalado automático es homogéneo y todas las instancias tienen la misma capacidad. Si esto no es así en el caso de su grupo, es posible que la capacidad prevista sea incorrecta. Por lo tanto, tenga cuidado cuando cree políticas de escalado predictivo para grupos de instancias mixtas, porque se pueden aprovisionar instancias de distintos tipos con una capacidad desigual. A continuación se presentan algunos ejemplos en los que la capacidad prevista será incorrecta:
  - La política de escalado predictivo se basa en la utilización de la CPU, pero el número de vCPU de cada instancia de Auto Scaling varía según los tipos de instancias.
  - La política de escalado predictivo se basa en la entrada o salida de la red, pero el rendimiento del ancho de banda de la red para cada instancia de Auto Scaling varía según los tipos de instancias. Por ejemplo, los tipos de instancias M5 y M5n son similares, pero el tipo de instancia M5n ofrece un rendimiento de red significativamente superior.

Regiones admitidas

- Este de EE. UU. (Norte de Virginia)
- Este de EE. UU. (Ohio)
- Oeste de EE. UU. (Norte de California)
- Oeste de EE. UU. (Oregón)
- África (Ciudad del Cabo)
- Asia-Pacífico (Hong Kong)
- Asia-Pacífico (Hyderabad)
- Asia-Pacífico (Yakarta)
- Asia-Pacífico (Melbourne)
- Asia-Pacífico (Mumbai)
- Asia-Pacífico (Osaka)
- Asia-Pacífico (Seúl)
- Asia-Pacífico (Singapur)
- Asia-Pacífico (Sídney)
- Asia-Pacífico (Tokio)
- Canadá (centro)
- Oeste de Canadá (Calgary)
- China (Pekín)

- China (Ningxia)
- Europa (Fráncfort)
- Europa (Irlanda)
- Europa (Londres)
- Europa (Milán)
- Europa (París)
- Europa (España)
- Europa (Estocolmo)
- Europa (Zúrich)
- Israel (Tel Aviv)
- Middle East (Bahrain)
- Medio Oriente (EAU)
- América del Sur (São Paulo)
- AWS GovCloud (US-East)
- AWS GovCloud (US-West)

Creación de una política de escalado predictivo para un grupo de escalado automático

Los siguientes procedimientos le ayudan a crear una política de escalado predictivo mediante el comando Consola de administración de AWS o AWS CLI.

Si el grupo de escalado automático es nuevo, debe proporcionar al menos 24 horas de datos antes de que Amazon EC2 Auto Scaling pueda generar una previsión para el grupo de escalado automático.

Contenido

- Creación de una política de escalado predictivo (consola)
- Cree una política de escalado predictivo (AWS CLI)

Creación de una política de escalado predictivo (consola)

Si es la primera vez que crea una política de escalado predictivo, se recomienda utilizar la consola para crear varias políticas de escalado predictivo en el modo de solo previsión. Esto le permite

poner a prueba los posibles efectos de diferentes métricas y valores de destino. Puede crear varias políticas de escalado predictivo para cada grupo de escalado automático, pero solo una de las políticas se puede utilizar para el escalado activo.

Creación de una política de escalado predictivo en la consola (métricas predefinidas)

Utilice el siguiente procedimiento para crear una política de escalado predictivo mediante métricas predefinidas (recuento de solicitudes de CPU I/O, red o Application Load Balancer por destino). La forma más simple de crear una política de escalado predictivo es mediante métricas predefinidas. Si prefiere utilizar métricas personalizadas en su lugar, consulte la Creación de una política de escalado predictivo en la consola (métricas personalizadas).

Para crear una política de escalado predictivo

1. Abra la consola Amazon EC2 en https://console.aws.amazon.com/ec2/y seleccione Auto Scaling Groups en el panel de navegación.
2. Seleccione la casilla situada junto al grupo de escalado automático.

   Se abre un panel dividido en la parte inferior de la página.
3. En la pestaña Automatic scaling (Escalado automático), en Scaling policies (Políticas de escalado), elija Create predictive scaling policy (Crear política de escalado automático).
4. Introduzca un nombre para la política.
5. Active Scale based on forecast (Escalado basado en pronóstico) para conceder a Amazon EC2 Auto Scaling permiso para empezar a escalar de inmediato.

   Para mantener la política en modo Forecast only (Solo pronóstico), mantenga Scale based on forecast (Escalado pasado en pronóstico) desactivado.
6. En Metrics (Métricas), elija las métricas de la lista de opciones. Las opciones incluyen CPU, Network In (Entrada de red), Network Out (Salida de red), Application Load Balancer request count (Recuento de solicitudes de Application Load Balancer) y Custom metric pair (Par de métricas personalizadas).

   Si eligió Application Load Balancer request count per target (Recuento de solicitudes de Application Load Balancer por destino), luego elija un grupo de destino en Target group (Grupo de destino). Application Load Balancer request count per target (Recuento de solicitudes de Application Load Balancer por destino) solo se admite si ha adjuntado un grupo de destino del Application Load Balancer al grupo de escalado automático.

Si eligió Custom metric pair (Par de métricas personalizadas), elija métricas individuales en las listas desplegables de Load metric (Métrica de carga) y Scaling metric (Métrica de escalado).

7. En Target utilization (Utilización de destino), ingrese el valor de destino que debe mantener Amazon EC2 Auto Scaling. Amazon EC2 Auto Scaling escala horizontalmente la capacidad hasta que la utilización media se encuentre en la utilización de destino, o hasta que alcance el número máximo de instancias que haya especificado.

Si la métrica de escalado es…	La utilización de destino representa…
CPU	Porcentaje de CPU que cada instancia debería utilizar idealmente.
Entrada de red	Número medio de bytes por minuto que cada instancia debería recibir idealmente.
Salida de red	Número medio de bytes por minuto que cada instancia debería enviar idealmente.
Recuento de solicitudes del Application Load Balancer por destino	Número medio de solicitudes por minuto que cada instancia debería recibir idealmente.

8. (Opcional) En el caso de Pre-launch las instancias, elija con qué antelación desea que se lancen las instancias antes de que la previsión indique un aumento de la carga.
9. (Opcional) En Max capacity behavior (Comportamiento de capacidad máxima), elija si desea permitir que Amazon EC2 Auto Scaling escale horizontalmente más allá de la capacidad máxima del grupo cuando la capacidad prevista supere el máximo definido. Activar esta configuración permite que el escalado horizontal se produzca durante los periodos en los que se prevé que el tráfico sea el máximo.
10. (Opcional) En Buffer maximum capacity above the forecasted capacity (Capacidad máxima del búfer por encima de la capacidad prevista, elija la capacidad adicional que se utilizará cuando la capacidad prevista se acerque o supere la capacidad máxima. El valor se especifica como porcentaje en relación con la capacidad prevista. Por ejemplo, si el búfer es 10, este valor indica un búfer del 10 por ciento. Por lo tanto, si la capacidad de previsión es 50 y la capacidad máxima es 40, la capacidad máxima nominal es 55.

Si se establece en 0, Amazon EC2 Auto Scaling puede escalar la capacidad por encima de la capacidad máxima para que sea igual a la capacidad máxima, pero no la supere.

11. Elija Create predictive scaling policy (Crear política de escalado predictivo).

Creación de una política de escalado predictivo en la consola (métricas personalizadas)

Utilice el procedimiento siguiente para crear una política de escalado predictivo mediante métricas personalizadas. Las métricas personalizadas pueden incluir otras métricas proporcionadas por usted CloudWatch o en las que publique CloudWatch. Para usar el recuento de solicitudes de CPU I/O, red o Application Load Balancer por destino, consulte. Creación de una política de escalado predictivo en la consola (métricas predefinidas)

Para crear una política de escalado predictivo mediante métricas personalizadas, debe hacer lo siguiente:

- Debe proporcionar las consultas sin procesar que permiten a Amazon EC2 Auto Scaling interactuar con las métricas CloudWatch internas. Para obtener más información, consulte Políticas avanzadas de escalado predictivo mediante métricas personalizadas. Para asegurarse de que Amazon EC2 Auto Scaling puede extraer los datos CloudWatch de las métricas, confirme que cada consulta devuelva puntos de datos. Confirme esto mediante la CloudWatch consola o la operación de la CloudWatch GetMetricDataAPI.

  > **Note**
  >
  > Proporcionamos cargas JSON de ejemplo en el editor JSON de la consola de Amazon EC2 Auto Scaling. Estos ejemplos te proporcionan una referencia de los pares clave-valor necesarios para añadir otras CloudWatch métricas proporcionadas AWS o en las que publicaste anteriormente. CloudWatch Puede utilizarlos como punto de partida y, luego, personalizarlos en función de sus necesidades.

- Si utiliza algún cálculo métrico, debe construir manualmente el JSON para que se ajuste a su escenario único. Para obtener más información, consulte Uso de expresiones de cálculos de métricas. Antes de utilizar las matemáticas métricas en su política, confirme que las consultas métricas basadas en expresiones matemáticas métricas son válidas y devuelven una única serie temporal. Confirme esto mediante la CloudWatch consola o la operación de la CloudWatch GetMetricDataAPI.

Si comete un error en una consulta al proporcionar datos incorrectos, como un nombre de grupo de escalado automático incorrecto, la previsión no tendrá ningún dato. Para solucionar problemas con las métricas personalizadas, consulte Consideraciones sobre la utilización de métricas personalizadas en una política de escalado predictivo.

Para crear una política de escalado predictivo

1. Abra la consola Amazon EC2 en https://console.aws.amazon.com/ec2/y seleccione Auto Scaling Groups en el panel de navegación.
2. Seleccione la casilla situada junto al grupo de escalado automático.

   Se abre un panel dividido en la parte inferior de la página.
3. En la pestaña Automatic scaling (Escalado automático), en Scaling policies (Políticas de escalado), elija Create predictive scaling policy (Crear política de escalado automático).
4. Introduzca un nombre para la política.
5. Active Scale based on forecast (Escalado basado en pronóstico) para conceder a Amazon EC2 Auto Scaling permiso para empezar a escalar de inmediato.

   Para mantener la política en modo Forecast only (Solo pronóstico), mantenga Scale based on forecast (Escalado pasado en pronóstico) desactivado.
6. En Metrics (Métricas), elija Custom metric pair (Par de métricas personalizadas).

   a. En Métrica de carga, elija CloudWatch Métrica personalizada para usar una métrica personalizada. Construcción de la carga JSON que contiene la definición de métricas de carga de la política y péguelo en el cuadro del editor JSON de modo que sustituya lo que ya se encuentra en el cuadro.

   b. En la métrica de escalado, elija CloudWatch Métrica personalizada para usar una métrica personalizada. Construcción de la carga JSON que contiene la definición de métricas de escalado de la política y péguelo en el cuadro del editor JSON de modo que sustituya lo que ya se encuentra en el cuadro.

   c. (Opcional) Para agregar una métrica de capacidad personalizada, active la casilla Add custom capacity metric (Agregar una métrica de capacidad personalizada). Construcción de la carga JSON que contiene la definición de métricas de capacidad de la política y péguelo en el cuadro del editor JSON de modo que sustituya lo que ya se encuentra en el cuadro.

Solo necesita habilitar esta opción para crear una nueva serie temporal de capacidad si los datos métricos de capacidad abarcan varios grupos de escalado automático. En este caso, debe utilizar las matemáticas métricas para agregar los datos en una sola serie temporal.

7. En Target utilization (Utilización de destino), ingrese el valor de destino que debe mantener Amazon EC2 Auto Scaling. Amazon EC2 Auto Scaling escala horizontalmente la capacidad hasta que la utilización media se encuentre en la utilización de destino, o hasta que alcance el número máximo de instancias que haya especificado.
8. (Opcional) En el caso de Pre-launch las instancias, elige con qué antelación quieres que se lancen las instancias antes de que la previsión indique que la carga aumentará.
9. (Opcional) En Max capacity behavior (Comportamiento de capacidad máxima), elija si desea permitir que Amazon EC2 Auto Scaling escale horizontalmente más allá de la capacidad máxima del grupo cuando la capacidad prevista supere el máximo definido. Activar esta configuración permite que el escalado horizontal se produzca durante los periodos en los que se prevé que el tráfico sea el máximo.
10. (Opcional) En Buffer maximum capacity above the forecasted capacity (Capacidad máxima del búfer por encima de la capacidad prevista, elija la capacidad adicional que se utilizará cuando la capacidad prevista se acerque o supere la capacidad máxima. El valor se especifica como porcentaje en relación con la capacidad prevista. Por ejemplo, si el búfer es 10, este valor indica un búfer del 10 por ciento. Por lo tanto, si la capacidad de previsión es 50 y la capacidad máxima es 40, la capacidad máxima nominal es 55.

    Si se establece en 0, Amazon EC2 Auto Scaling puede escalar la capacidad por encima de la capacidad máxima para que sea igual a la capacidad máxima, pero no la supere.
11. Elija Create predictive scaling policy (Crear política de escalado predictivo).

Cree una política de escalado predictivo (AWS CLI)

Utilice AWS CLI lo siguiente para configurar las políticas de escalado predictivo para su grupo de Auto Scaling. Reemplace cada *`user input placeholder`* por su propia información.

Para obtener más información sobre las CloudWatch métricas que puede especificar, consulte la referencia PredictiveScalingMetricSpecificationde la API Amazon EC2 Auto Scaling.

Ejemplo 1: Política de escalado predictivo que crea pronósticos, pero no escala

En la siguiente política de ejemplo se muestra una configuración de política completa que utiliza métricas de utilización de CPU para el escalado predictivo con una utilización de destino de`40`. `ForecastOnly` se utiliza de forma predeterminada, a menos que especifique explícitamente qué modo utilizar. Guarde esta configuración en un archivo llamado `config.json`.

```
{
    "MetricSpecifications": [
        {
            "TargetValue": 40,
            "PredefinedMetricPairSpecification": {
                "PredefinedMetricType": "ASGCPUUtilization"
            }
        }
    ]
}
```

Para crear la política de la línea de comando, ejecute el comando put-scaling-policy con el archivo de configuración especificado, tal y como se muestra en el ejemplo siguiente.

```
aws autoscaling put-scaling-policy --policy-name cpu40-predictive-scaling-policy \
  --auto-scaling-group-name my-asg --policy-type PredictiveScaling \
  --predictive-scaling-configuration file://config.json
```

Si se ejecuta correctamente, este comando devuelve el nombre de recurso de Amazon (ARN) de la política.

```
{
  "PolicyARN": "arn:aws:autoscaling:region:account-id:scalingPolicy:2f4f5048-d8a8-4d14-
b13a-d1905620f345:autoScalingGroupName/my-asg:policyName/cpu40-predictive-scaling-
policy",
  "Alarms": []
}
```

Ejemplo 2: Política de escalado predictivo que pronostica y escala

Para una política que permita a Amazon EC2 Auto Scaling pronosticar y escalar, agregue la propiedad `Mode` con un valor de `ForecastAndScale`. En el ejemplo siguiente se muestra la configuración de una política que utiliza métricas de recuento de solicitudes del Application Load

Balancer. La utilización de destino es `1000`, y el escalado predictivo se establece en el modo `ForecastAndScale`.

```
{
    "MetricSpecifications": [
        {
            "TargetValue": 1000,
            "PredefinedMetricPairSpecification": {
                "PredefinedMetricType": "ALBRequestCount",
                "ResourceLabel": "app/my-alb/778d41231b141a0f/targetgroup/my-alb-
target-group/943f017f100becff"
            }
        }
    ],
    "Mode": "ForecastAndScale"
}
```

Para crear esta política, ejecute el comando put-scaling-policy con el archivo de configuración especificado, tal y como se muestra en el ejemplo siguiente.

```
aws autoscaling put-scaling-policy --policy-name alb1000-predictive-scaling-policy \
  --auto-scaling-group-name my-asg --policy-type PredictiveScaling \
  --predictive-scaling-configuration file://config.json
```

Si se ejecuta correctamente, este comando devuelve el nombre de recurso de Amazon (ARN) de la política.

```
{
  "PolicyARN": "arn:aws:autoscaling:region:account-
id:scalingPolicy:19556d63-7914-4997-8c81-d27ca5241386:autoScalingGroupName/my-
asg:policyName/alb1000-predictive-scaling-policy",
  "Alarms": []
}
```

Ejemplo 3: Política de escalado predictivo que puede escalar por encima de la capacidad máxima

En el ejemplo siguiente se muestra cómo crear una política que puede escalar más allá del límite de tamaño máximo del grupo cuando se necesita para gestionar una carga superior a la normal. De forma predeterminada, Amazon EC2 Auto Scaling no escala la capacidad de EC2 por encima de la capacidad máxima definida. Sin embargo, podría resultar útil permitir que escale la capacidad un poco más allá para evitar problemas de rendimiento o disponibilidad.

Para dejar espacio para que Amazon EC2 Auto Scaling aprovisione la capacidad adicional cuando se prevea que la capacidad sea igual o muy cercana al tamaño máximo de su grupo, especifique las propiedades `MaxCapacityBreachBehavior` y `MaxCapacityBuffer`, como se muestra en el ejemplo siguiente. Debe especificar `MaxCapacityBreachBehavior` con un valor de `IncreaseMaxCapacity`. El número máximo de instancias que el grupo puede tener depende del valor de `MaxCapacityBuffer`.

```
{
    "MetricSpecifications": [
        {
            "TargetValue": 70,
            "PredefinedMetricPairSpecification": {
                "PredefinedMetricType": "ASGCPUUtilization"
            }
        }
    ],
    "MaxCapacityBreachBehavior": "IncreaseMaxCapacity",
    "MaxCapacityBuffer": 10
}
```

En este ejemplo, la política está configurada para utilizar un búfer del 10 por ciento (`"MaxCapacityBuffer": 10`), de forma que si la capacidad prevista es de 50 y la capacidad máxima es de 40, la capacidad máxima efectiva es de 55. Una política que pueda escalar la capacidad por encima de la capacidad máxima para que sea igual a la capacidad prevista, pero no mayor, tendría un búfer de 0 (`"MaxCapacityBuffer": 0`).

Para crear esta política, ejecute el comando put-scaling-policy con el archivo de configuración especificado, tal y como se muestra en el ejemplo siguiente.

```
aws autoscaling put-scaling-policy --policy-name cpu70-predictive-scaling-policy \
  --auto-scaling-group-name my-asg --policy-type PredictiveScaling \
  --predictive-scaling-configuration file://config.json
```

Si se ejecuta correctamente, este comando devuelve el nombre de recurso de Amazon (ARN) de la política.

```
{
  "PolicyARN": "arn:aws:autoscaling:region:account-id:scalingPolicy:d02ef525-8651-4314-
bf14-888331ebd04f:autoScalingGroupName/my-asg:policyName/cpu70-predictive-scaling-
policy",
```

```
  "Alarms": []
}
```

Evaluación de las políticas de escalado predictivo

Antes de utilizar una política de escalado predictivo para escalar automáticamente su grupo de escalado automático, consulte las recomendaciones y otros datos de la política en la consola de Amazon EC2 Auto Scaling. Esto es importante porque no es recomendable que una política de escalado predictivo amplíe su capacidad real hasta que sepa que sus predicciones son precisas.

Si el grupo de escalado automático es nuevo, espere 24 horas para que Amazon EC2 Auto Scaling cree la primera previsión.

Cuando Amazon EC2 Auto Scaling crea una previsión, utiliza datos históricos. Si su grupo de escalado automático aún no cuenta con muchos datos históricos recientes, Amazon EC2 Auto Scaling podría rellenar temporalmente la previsión con agregados creados a partir de los agregados históricos disponibles actualmente. Las previsiones se rellenan hasta dos semanas antes de la fecha de creación de la política.

Contenido

- Visualización de las recomendaciones de escalado predictivo
- Revisión de los gráficos de supervisión del escalado predictivo
- Supervise las métricas de escalado predictivo con CloudWatch

Visualización de las recomendaciones de escalado predictivo

Para poder llevar a cabo un análisis eficaz, Amazon EC2 Auto Scaling debe tener al menos dos políticas de escalado predictivo para comparar. (Sin embargo, aún puede revisar los resultados de una sola política). Al crear varias políticas, puede evaluar una política que usa una métrica en comparación con una política que usa una diferente. También puede evaluar el impacto de diferentes combinaciones de valores y métricas de destino. Una vez creadas las políticas de escalado predictivo, Amazon EC2 Auto Scaling comienza inmediatamente a evaluar qué política haría un mejor trabajo a la hora de escalar el grupo.

Visualización de las recomendaciones en la consola de Amazon EC2 Auto Scaling

1. Abra la consola Amazon EC2 en https://console.aws.amazon.com/ec2/y seleccione Auto Scaling Groups en el panel de navegación.

2. Seleccione la casilla situada junto al grupo de escalado automático.

   Se abre un panel dividido en la parte inferior de la página.

3. En la pestaña Escalado automático, en Políticas de escalado predictivo, puede ver los detalles de una política junto con nuestra recomendación. La recomendación indica si la política de escalado predictivo funciona mejor que si no se utiliza.

   Si no está seguro de si una política de escalado predictivo es adecuada para su grupo, revise las columnas Impacto en la disponibilidad e Impacto en los costos para elegir la política correcta. La información de cada columna indica cuál es el impacto de la política.

   - Impacto en la disponibilidad: describe si la política evitaría un impacto negativo en la disponibilidad al aprovisionar suficientes instancias para gestionar la carga de trabajo, en comparación con no utilizar la política.
   - Impacto en los costos: describe si la política evitaría un impacto negativo en los costos al no aprovisionar en exceso las instancias, en comparación con no utilizar la política. Al aprovisionar demasiado, las instancias quedan infrautilizadas o inactivas, lo que no hace más que aumentar el impacto en los costos.

   Si tiene varias políticas, aparecerá una etiqueta que pondrá Mejor predicción junto al nombre de la política que ofrece la mayor cantidad de beneficios de disponibilidad a un costo menor. Se da más importancia al impacto en la disponibilidad.

4. (Opcional) Para seleccionar el periodo de tiempo deseado de los resultados de las recomendaciones, elija el valor que prefiera en el menú desplegable Periodo de evaluación: 2 días, 1 semana, 2 semanas, 4 semanas, 6 semanas u 8 semanas. De forma predeterminada, el periodo de evaluación son las dos últimas semanas. Un periodo de evaluación más largo proporciona más puntos de datos para los resultados de la recomendación. Sin embargo, es posible que agregar más puntos de datos no mejore los resultados si los patrones de carga han cambiado, por ejemplo, después de un periodo de demanda excepcional. En este caso, puede obtener una recomendación más específica si consulta datos más recientes.

> **Note**
>
> Las recomendaciones se generan solo para las políticas que están en el modo Solo previsión. La característica de recomendaciones funciona mejor cuando una política está en el modo Solo previsión durante el periodo de evaluación. Si inicia una política en

modo Previsión y escalado y luego la cambia al modo Solo previsión, es probable que los resultados de esa política estén sesgados. Esto se debe a que la política ya ha contribuido a la capacidad real.

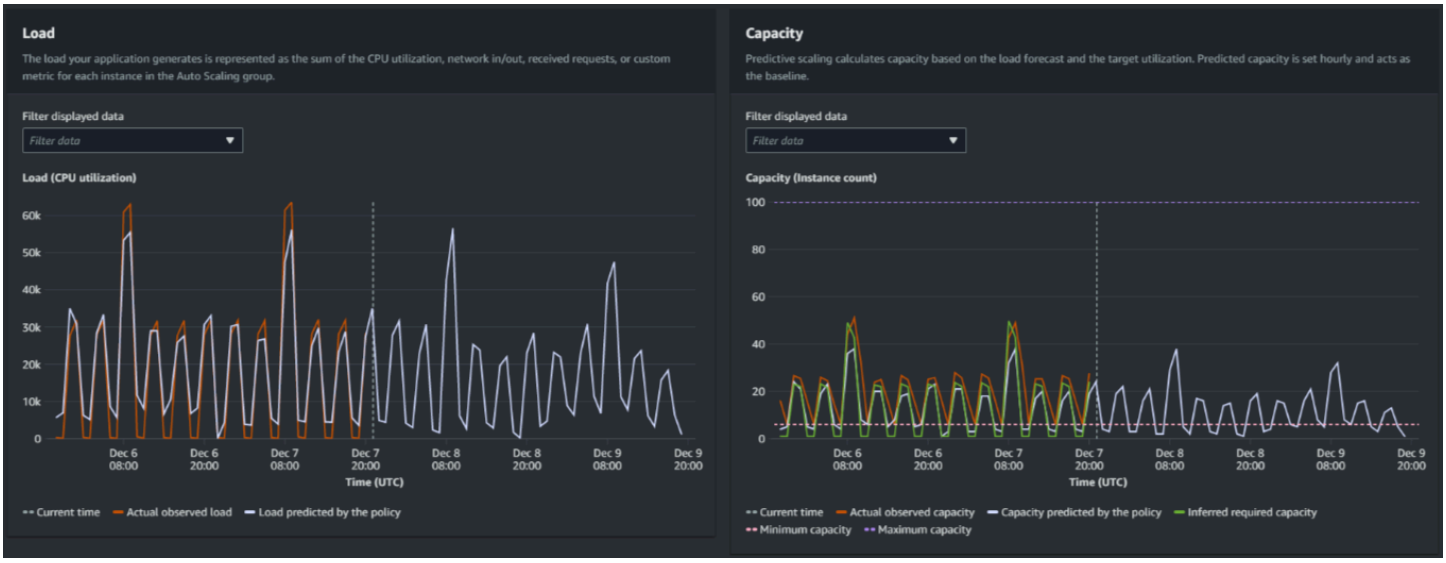
Revisión de los gráficos de supervisión del escalado predictivo

En la consola de Amazon EC2 Auto Scaling, puede revisar la previsión de los días, semanas o meses anteriores para visualizar el rendimiento de la política a lo largo del tiempo. También puede utilizar esta información para evaluar la precisión de las predicciones a la hora de decidir si va a permitir que una política amplíe su capacidad real.

Visualización de los gráficos de supervisión en la consola de Amazon EC2 Auto Scaling

1. Elija una política de la lista Políticas de escalado predictivo.
2. En la sección Supervisión, puede ver las previsiones pasadas y futuras de su política de carga y capacidad y compararlas con los valores reales. En el gráfico Carga se muestra el pronóstico de carga y los valores reales para la métrica de carga que elija. En el gráfico Capacidad se muestra el número de instancias pronosticado por la política. También se incluye el número real de instancias lanzadas. La línea vertical separa los valores históricos de las previsiones futuras. Estos gráficos estarán disponibles poco después de que se cree la política.
3. (Opcional) Para cambiar la cantidad de datos históricos que se muestra en el gráfico, elija su valor preferido en el menú desplegable Periodo de evaluación en la parte superior de la página. El periodo de evaluación no transforma los datos de esta página de ninguna manera. Solo cambia la cantidad de datos históricos que se muestran.

La siguiente imagen muestra los gráficos Carga y Capacidad cuando las previsiones se han aplicado varias veces. El escalado predictivo prevé la carga en función de los datos de carga históricos. La carga que genera la aplicación se representa como la suma del uso de la CPU, la red in/out, las solicitudes recibidas o la métrica personalizada de cada instancia del grupo Auto Scaling. El escalado predictivo calcula las necesidades de capacidad futuras en función de la previsión de carga y el uso objetivo que se quiera alcanzar en la métrica de escalado.



Compare los datos en el Cargar gráfico

Cada línea horizontal representa un conjunto diferente de puntos de datos de los que se ha informado en intervalos de una hora:

1. La carga real observada utiliza la estadística SUM de la métrica de carga elegida para mostrar la carga total por hora en el pasado.
2. Carga pronosticada por la política muestra la predicción de carga por hora. Esta predicción se basa en las dos semanas anteriores de observaciones de carga reales.

Compare los datos en el Capacidad gráfico

Cada línea horizontal representa un conjunto diferente de puntos de datos de los que se ha informado en intervalos de una hora:

1. La capacidad observada real muestra la capacidad real pasada de su grupo de Auto Scaling cuando la GroupTotalInstancesmétrica está habilitada. Esta capacidad depende de las demás políticas de escalado y del tamaño mínimo del grupo durante el periodo de tiempo seleccionado.
2. Capacidad pronosticada por la política muestra la capacidad de referencia que puede esperar tener al principio de cada hora cuando la política esté en modo Previsión y escalado.
3. Capacidad necesaria inferida muestra la capacidad ideal para mantener la métrica de escalado en el valor objetivo que haya elegido.
4. Capacidad mínima muestra la capacidad mínima del grupo de escalado automático.
5. Capacidad máxima muestra la capacidad máxima del grupo de escalado automático.

Para calcular la capacidad necesaria inferida, primero asumimos que cada instancia se utiliza por igual en un valor objetivo específico. En la práctica, las instancias no se utilizan por igual. Sin embargo, si asumimos que la utilización se distribuye de manera uniforme entre las instancias, podemos hacer una estimación probabilística de la cantidad de capacidad que se necesita. A continuación, se calcula el requisito de capacidad para que sea inversamente proporcional a la métrica de escalado que se utilizó para la política de escalado predictivo. En otras palabras, a medida que aumenta la capacidad, la métrica de escalado disminuye al mismo ritmo. Por ejemplo, si la capacidad se duplica, la métrica de escalado debe reducirse a la mitad.

La fórmula para la capacidad necesaria inferida:

```
sum of (actualCapacityUnits*scalingMetricValue)/(targetUtilization)
```

Por ejemplo, tomamos los valores de `actualCapacityUnits` (`10`) y `scalingMetricValue` (`30`) de una hora determinada. A continuación, tomamos el valor de `targetUtilization` que especificó en su política de escalado predictivo (`60`) y calculamos la capacidad necesaria inferida de la misma hora. Esto devuelve un valor de `5`. Esto significa que cinco es la cantidad de capacidad inferida necesaria para mantener la capacidad en proporción inversa directa al valor de destino de la métrica de escalado.

> **Note**
>
> Hay varias palancas disponibles para ajustar y mejorar el ahorro de costos y la disponibilidad de la aplicación.
>
> - Se utiliza el escalado predictivo para la capacidad de referencia y el escalado dinámico para gestionar la capacidad adicional. El escalado dinámico funciona independientemente del escalado predictivo, escalando vertical u horizontalmente en función de la utilización actual. En primer lugar, Amazon EC2 Auto Scaling calcula el número recomendado de instancias de cada política de escalado dinámico. A continuación, se escala en función de la política que proporciona la mayor cantidad de instancias.
> - Para permitir que se reduzca horizontalmente cuando la carga disminuya, su grupo de escalado automático siempre debe tener al menos una política de escalado dinámico con la parte de reducción horizontal habilitada.
> - Puede mejorar el rendimiento del escalado si se asegura de que su capacidad mínima y máxima no sean demasiado restrictivas. Se impedirá que una política con un número recomendado de instancias que no se encuentre dentro del rango de capacidad mínima y máxima se reduzca o escale horizontalmente.

Supervise las métricas de escalado predictivo con CloudWatch

Según sus necesidades, es posible que prefiera acceder a los datos de monitoreo para el escalado predictivo desde Amazon CloudWatch en lugar de desde la consola Amazon EC2 Auto Scaling. Después de crear una política de escalado predictivo, la política recopila datos que se utilizan para pronosticar su carga y capacidad futuras. Una vez recopilados estos datos, se almacenan automáticamente a CloudWatch intervalos regulares. A continuación, puede utilizarlos CloudWatch para visualizar el rendimiento de la política a lo largo del tiempo. También puede crear CloudWatch alarmas que le notifiquen cuando los indicadores de rendimiento cambien más allá de los límites que usted haya definido CloudWatch.

Temas

- Visualización de los datos de las previsiones
- Creación de métricas de precisión mediante la matemática métrica

Visualización de los datos de las previsiones

Puede ver los datos de previsión de carga y capacidad para una política de escalado predictivo en CloudWatch. Esto puede resultar útil a la hora de visualizar las previsiones comparándolas con otras CloudWatch métricas en un único gráfico. También puede ser útil cuando desee ver un intervalo de tiempo mayor para poder identificar las tendencias a lo largo del tiempo. Puede acceder a métricas históricas de hasta 15 meses para obtener una mejor perspectiva del rendimiento de su política.

Para obtener más información, consulte Dimensiones y métricas de escalado predictivo.

Para ver los datos históricos de las previsiones mediante la consola CloudWatch

1. Abra la CloudWatch consola en https://console.aws.amazon.com/cloudwatch/.
2. En el panel de navegación, elija Metrics (Métricas) y, a continuación, All metrics (Todas las métricas).
3. Elija el espacio de nombre de métrica Auto Scaling (Escalado automático).
4. Elija una de las siguientes opciones para ver las métricas de previsión de carga o de previsión de capacidad:
   - Predictive Scaling Load Forecasts (Pronósticos de carga de escala predictiva)
   - Predictive Scaling Capacity Forecasts (Pronósticos de capacidad de escalabilidad predictiva)

5. En el campo de búsqueda, ingrese el nombre de la política de escalado predictivo o el nombre del grupo de escalado automático y, a continuación, pulse Intro para filtrar los resultados.
6. Para representar gráficamente una métrica, active la casilla de verificación situada junto a ella. Para cambiar el nombre del gráfico, seleccione el icono de lápiz. Para cambiar el intervalo de tiempo, seleccione uno de los valores predefinidos o elija custom (personalizado). Para obtener más información, consulta Cómo graficar una métrica en la Guía del CloudWatch usuario de Amazon.
7. Para cambiar la estadística, elija la pestaña Graphed metrics. Elija el encabezado de columna o un valor individual y, a continuación, elija una estadística diferente. Si bien puedes elegir cualquier estadística para cada métrica, no todas las estadísticas son útiles para PredictiveScalingLoadForecastlas PredictiveScalingCapacityForecastmétricas. Por ejemplo, las estadísticas Promedio, Mínimo y Máximo son útiles, pero la estadística Suma no lo es.
8. Para agregar otra métrica al gráfico, en Browse (Examinar), elija All (Todo), busque la métrica específica y luego seleccione la casilla de verificación que aparece a su lado. Puede añadir hasta 10 métricas.

   Por ejemplo, para agregar los valores reales de uso de la CPU al gráfico, elija el espacio de nombres de EC2 y, a continuación, elija By Auto Scaling Group (Por grupo de escalado automático). A continuación, seleccione la casilla de verificación de la métrica CPUUtilization y el grupo de escalado automático específico.
9. (Opcional) Para añadir el gráfico a un CloudWatch panel, elija Acciones y Añadir al panel.

Creación de métricas de precisión mediante la matemática métrica

Con la matemática métrica, puede consultar múltiples CloudWatch métricas y usar expresiones matemáticas para crear nuevas series temporales basadas en estas métricas. Puede visualizar las series temporales resultantes en la CloudWatch consola y añadirlas a los paneles. Para obtener más información sobre las matemáticas métricas, consulte Uso de las matemáticas métricas en la Guía del CloudWatch usuario de Amazon.

Con la matemática métrica, puede representar gráficamente los datos que Amazon EC2 Auto Scaling genera para reducir horizontalmente de diferentes maneras. Esto es de utilidad para supervisar el rendimiento de las políticas a lo largo del tiempo y a comprender si se puede mejorar la combinación de métricas.

Por ejemplo, puede usar una expresión matemática métrica para supervisar el mean absolute percentage error (error porcentual absoluto medio o MAPE). La métrica MAPE ayuda a supervisar

la diferencia entre los valores pronosticados y los valores reales observados durante un periodo de previsión determinado. Los cambios en el valor de MAPE pueden indicar si el rendimiento de la política se degrada con el tiempo a medida que cambia la naturaleza de la aplicación. Un aumento en MAPE indica una brecha más amplia entre los valores pronosticados y los valores reales.

Ejemplo: expresiones matemáticas de métricas

Para empezar a utilizar este tipo de gráfica, puede crear una expresión matemática métrica como la que se muestra en el siguiente ejemplo.

```
{
  "MetricDataQueries": [
    {
      "Expression": "TIME_SERIES(AVG(ABS(m1-m2)/m1))",
      "Id": "e1",
      "Period": 3600,
      "Label": "MeanAbsolutePercentageError",
      "ReturnData": true
    },
    {
      "Id": "m1",
      "Label": "ActualLoadValues",
      "MetricStat": {
        "Metric": {
          "Namespace": "AWS/EC2",
          "MetricName": "CPUUtilization",
          "Dimensions": [
            {
              "Name": "AutoScalingGroupName",
              "Value": "my-asg"
            }
          ]
        },
        "Period": 3600,
        "Stat": "Sum"
      },
      "ReturnData": false
    },
    {
      "Id": "m2",
      "Label": "ForecastedLoadValues",
      "MetricStat": {
        "Metric": {
```

```
          "Namespace": "AWS/AutoScaling",
          "MetricName": "PredictiveScalingLoadForecast",
          "Dimensions": [
            {
              "Name": "AutoScalingGroupName",
              "Value": "my-asg"
            },
            {
              "Name": "PolicyName",
              "Value": "my-predictive-scaling-policy"
            },
            {
              "Name": "PairIndex",
              "Value": "0"
            }
          ]
        },
        "Period": 3600,
        "Stat": "Average"
      },
      "ReturnData": false
    }
  ]
}
```

En lugar de una sola métrica, hay una matriz de estructuras de consulta de datos métricos para `MetricDataQueries`. Cada elemento de `MetricDataQueries` obtiene una métrica o realiza una expresión matemática. El primer elemento, `e1`, es la expresión matemática. La expresión designada establece el parámetro `ReturnData` a `true`, que en última instancia produce una sola serie temporal. Para todas las demás métricas, el valor `ReturnData` es `false`.

En el ejemplo, la expresión designada utiliza los valores reales y previstos como entrada y devuelve la nueva métrica (MAPE). `m1`es la CloudWatch métrica que contiene los valores de carga reales (suponiendo que la utilización de la CPU sea la métrica de carga que se especificó originalmente para la política denominada`my-predictive-scaling-policy`). `m2`es la CloudWatch métrica que contiene los valores de carga previstos. La sintaxis matemática de la métrica MAPE es la siguiente:

Average of (abs ((Actual - Forecast)/(Actual))) (Promedio de (abs ((Real - Previsión)/(Real))))

Visualización de métricas de precisión y configuración de alarmas

Para visualizar los datos de las métricas de precisión, seleccione la pestaña Métricas de la CloudWatch consola. Puede hacer una representación gráfica de los datos desde allí. Para obtener más información, consulta Cómo añadir una expresión matemática a un CloudWatch gráfico en la Guía del CloudWatch usuario de Amazon.

Puede configurar una alarma para una métrica que supervise desde la sección Metrics. Mientras está en la pestaña Métricas diagramadas, puede seleccionar el icono Crear alarma en la columna Acciones. El icono Create alarm se representa como una pequeña campana. Para obtener más información y opciones de notificación, consulte Crear una CloudWatch alarma basada en una expresión matemática métrica y Notificar a los usuarios los cambios de alarma en la Guía del CloudWatch usuario de Amazon.

Como alternativa, puede usar GetMetricDatay PutMetricAlarmrealizar cálculos mediante cálculos métricos y crear alarmas en función de los resultados.

Anulación de valores de pronóstico mediante acciones programadas

A veces, es posible que tenga información adicional sobre los requisitos futuros de la aplicación que el cálculo del pronóstico no pueda tener en cuenta. Por ejemplo, los cálculos de pronóstico podrían subestimar la capacidad necesaria para un próximo evento de marketing. Puede utilizar acciones programadas para anular temporalmente el pronóstico durante periodos futuros. Las acciones programadas se pueden ejecutar de forma periódica, o en una fecha y hora específicas cuando hay fluctuaciones de demanda únicas.

Por ejemplo, puede crear una acción programada con una capacidad mínima superior a la pronosticada. En tiempo de ejecución, Amazon EC2 Auto Scaling actualiza la capacidad mínima del grupo de escalado automático. Dado que el escalado predictivo optimiza la capacidad, se cumple una acción programada con una capacidad mínima superior a los valores del pronóstico. Esto evita que la capacidad sea menor que lo esperado. Para dejar de anular el pronóstico, utilice una segunda acción programada para devolver la capacidad mínima a su configuración original.

En el siguiente procedimiento se describen los pasos para anular el pronóstico durante periodos futuros.

Temas

- Paso 1: (opcional) Analizar los datos de serie temporal
- Paso 2: Crear dos acciones programadas

> **Important**
>
> En este tema, se supone que está intentando anular la previsión para escalar a una capacidad superior a la prevista. Si necesita reducir temporalmente la capacidad sin que interfiera una política de escalado predictivo, utilice el modo de solo previsión. Mientras esté en el modo de solo previsión, el escalado predictivo seguirá generando previsiones, pero no aumentará automáticamente la capacidad. De esta manera, puede supervisar la utilización de los recursos y reducir manualmente el tamaño del grupo según sea necesario. Para obtener más información acerca del escalado manual, consulte Escalado manual para Amazon EC2 Auto Scaling.

Paso 1: (opcional) Analizar los datos de serie temporal

Para comenzar, analice los datos de serie temporal del pronóstico. Este es un paso opcional, pero resulta útil si desea comprender los detalles del pronóstico.

1. Recuperar el pronóstico

   Una vez creado el pronóstico, puede consultar un periodo específico en el pronóstico. El objetivo de la consulta es obtener una vista completa de los datos de serie temporal para un periodo específico.

   La consulta puede incluir hasta dos días de datos de pronósticos futuros. Si hace tiempo utiliza el escalado predictivo, también puede acceder a los datos de pronóstico anteriores. Sin embargo, la duración máxima entre la hora de inicio y la hora de finalización es de 30 días.

   Para obtener la previsión mediante el comando get-predictive-scaling-forecast, introduzca los siguientes parámetros en AWS CLI el comando:

   - Ingrese el nombre del grupo de escalado automático en el parámetro `--auto-scaling-group-name`.
   - Ingrese el nombre de la política en el parámetro `--policy-name`.
   - Indique la hora de inicio en el parámetro `--start-time` para devolver solo los datos pronosticados para la hora especificada o con posterioridad.
   - Indique la hora de finalización en el parámetro `--end-time` para devolver solo los datos pronosticados para antes de la hora especificada.

```
aws autoscaling get-predictive-scaling-forecast --auto-scaling-group-name my-asg \
    --policy-name cpu40-predictive-scaling-policy \
    --start-time "2021-05-19T17:00:00Z" \
    --end-time "2021-05-19T23:00:00Z"
```

Si se ejecuta correctamente, el comando devuelve datos similares al ejemplo siguiente.


```
{
    "LoadForecast": [
        {
            "Timestamps": [
                "2021-05-19T17:00:00+00:00",
                "2021-05-19T18:00:00+00:00",
                "2021-05-19T19:00:00+00:00",
                "2021-05-19T20:00:00+00:00",
                "2021-05-19T21:00:00+00:00",
                "2021-05-19T22:00:00+00:00",
                "2021-05-19T23:00:00+00:00"
            ],
            "Values": [
                153.0655799339254,
                128.8288551285919,
                107.1179447150675,
                197.3601844551528,
                626.4039934516954,
                596.9441277518481,
                677.9675713779869
            ],
            "MetricSpecification": {
                "TargetValue": 40.0,
                "PredefinedMetricPairSpecification": {
                    "PredefinedMetricType": "ASGCPUUtilization"
                }
            }
        }
    ],
    "CapacityForecast": {
        "Timestamps": [
            "2021-05-19T17:00:00+00:00",
            "2021-05-19T18:00:00+00:00",
            "2021-05-19T19:00:00+00:00",
```

```
            "2021-05-19T20:00:00+00:00",
            "2021-05-19T21:00:00+00:00",
            "2021-05-19T22:00:00+00:00",
            "2021-05-19T23:00:00+00:00"
        ],
        "Values": [
            2.0,
            2.0,
            2.0,
            2.0,
            4.0,
            4.0,
            4.0
        ]
    },
    "UpdateTime": "2021-05-19T01:52:50.118000+00:00"
}
```


La respuesta incluye dos pronósticos: `LoadForecast` y `CapacityForecast`. `LoadForecast` muestra el pronóstico de carga por hora. `CapacityForecast` muestra los valores pronosticados para la capacidad que se necesita cada hora para gestionar la carga pronosticada mientras se mantiene un `TargetValue` de 40,0 (una utilización media de la CPU del 40 %).

2. Identifique el periodo de destino

   Identifique la o las horas en las que debe tener lugar la fluctuación de la demanda única. Recuerde que las fechas y horas que aparecen en el pronóstico están en UTC.

Paso 2: Crear dos acciones programadas

A continuación, cree dos acciones programadas para un periodo específico en el que la aplicación tendrá una carga superior a la pronosticada. Por ejemplo, si tiene un evento de marketing que incrementará el tráfico hacia su sitio durante un periodo de tiempo limitado, puede programar una acción única para actualizar la capacidad mínima cuando comience. A continuación, programe otra acción para devolver la capacidad mínima a la configuración original cuando el evento finalice.

Para crear dos acciones programadas para eventos únicos (consola)

1. Abra la consola Amazon EC2 en https://console.aws.amazon.com/ec2/y seleccione Auto Scaling Groups en el panel de navegación.
2. Seleccione la casilla situada junto al grupo de escalado automático.

Se abre un panel dividido en la parte inferior de la página.

3. En la pestaña Escalado automático, en Acciones programadas, elija Crear acción programada.
4. Rellene los siguientes ajustes para la acción configuración:

   a. Ingrese un Name (Nombre) para la acción programada.

   b. Para Min (Mínimo), ingrese la nueva capacidad mínima para el grupo de escalado automático. Min debe ser menor o igual que el tamaño máximo del grupo. Si el nuevo valor de Min es mayor que el tamaño máximo del grupo, debe actualizar Max (Máximo).

   c. En Recurrencia, elija Una vez.

   d. En Time zone (Zona horaria), elija una zona horaria. Si no se elige ninguna zona horaria, se utiliza ETC/UTC de forma predeterminada.

   e. Defina Specific start time (Hora de inicio específica).
5. Seleccione Crear.

   La consola muestra las acciones programadas para el grupo de escalado automático.
6. Configure una segunda acción programada para que la capacidad mínima recupere la configuración original al final del evento. El escalado predictivo puede escalar la capacidad únicamente cuando el valor establecido para Min (Mínimo) es menor que los valores del pronóstico.

Para crear dos acciones programadas para eventos únicos (AWS CLI)

Para utilizar el AWS CLI para crear las acciones programadas, utilice el comando put-scheduled-update-group-action.

Por ejemplo, definamos una programación que mantenga una capacidad mínima de tres instancias el 19 de mayo a las 17:00 durante ocho horas. En los siguientes comandos se muestra cómo implementar este escenario.

El primer comando put-scheduled-update-group-action indica a Amazon EC2 Auto Scaling que actualice la capacidad mínima del grupo de escalado automático especificado a las 17:00 (UTC) del 19 de mayo de 2021.

```
aws autoscaling put-scheduled-update-group-action --scheduled-action-name my-event-
start \
```

```
  --auto-scaling-group-name my-asg --start-time "2021-05-19T17:00:00Z" --minimum-
capacity 3
```

El segundo comando indica a Amazon EC2 Auto Scaling que establezca la capacidad mínima del grupo en uno a la 1:00 (UTC) del 20 de mayo de 2021.

```
aws autoscaling put-scheduled-update-group-action --scheduled-action-name my-event-end
 \
  --auto-scaling-group-name my-asg --start-time "2021-05-20T01:00:00Z" --minimum-
capacity 1
```

Después de agregar estas acciones programadas al grupo de escalado automático, Amazon EC2 Auto Scaling realiza lo siguiente:

- A las 17:00 (UTC) del 19 de mayo de 2021, se ejecuta la primera acción programada. Si el grupo tiene actualmente menos de tres instancias, el grupo se escala horizontalmente hasta tres instancias. Durante este tiempo y durante las siguientes ocho horas, Amazon EC2 Auto Scaling puede seguir escalando horizontalmente si la capacidad prevista es superior a la capacidad real o si existe una política de escalado dinámico en vigor.
- A la 1:00 (UTC) del 20 de mayo de 2021, se ejecuta la segunda acción programada. Esto devuelve la capacidad mínima a la configuración original al final del evento.

Escalado basado en programaciones recurrentes

Para anular el pronóstico para el mismo periodo cada semana, cree dos acciones programadas y proporcione la lógica de fecha y hora utilizando una expresión cron.

El formato de expresión cron consta de cinco campos separados por espacios: [Minuto] [Hora] [] []. Day_of_Month Month_of_Year Day_of_Week Los campos pueden contener cualquier valor permitido, incluidos caracteres especiales.

Por ejemplo, la siguiente expresión cron ejecuta una acción todos los martes a las 6:30. El asterisco se utiliza como comodín para coincidir con todos los valores de un campo.

```
30 6 * * 2
```

Véase también

Para obtener más información acerca de cómo crear, enumerar, editar y eliminar acciones programadas, consulte Escalado programado para Amazon EC2 Auto Scaling.

Políticas avanzadas de escalado predictivo mediante métricas personalizadas

En una política de escalado predictivo, puede utilizar métricas predefinidas o personalizadas. Las métricas personalizadas son útiles cuando las métricas predefinidas (CPU I/O, red y recuento de solicitudes de Application Load Balancer) no describen suficientemente la carga de la aplicación.

Al crear una política de escalado predictivo con métricas personalizadas, puede especificar otras CloudWatch métricas proporcionadas por AWS, o bien puede especificar métricas que defina y publique usted mismo. También puede utilizar las matemáticas métricas para agregar y transformar las métricas existentes en una nueva serie temporal que AWS no se realice un seguimiento automático. Cuando combina valores en los datos, por ejemplo, al calcular nuevas sumas o promedios, se denomina aggregating (agrupando). Los datos obtenidos se denominan aggregate (agrupación).

En la siguiente sección se encuentran las mejores prácticas y ejemplos de cómo construir la estructura JSON para la política.

Temas

- Prácticas recomendadas
- Requisitos previos
- Construir JSON para métricas personalizadas
- Consideraciones sobre la utilización de métricas personalizadas en una política de escalado predictivo
- Limitaciones

Prácticas recomendadas

Las siguientes prácticas recomendadas pueden ayudarlo a utilizar las métricas personalizadas de manera más eficaz:

- Para la especificación de las métricas de carga, la métrica que resulta de mayor utilidad es la que representa la carga en un grupo de escalado automático en su conjunto, independientemente de la capacidad del grupo.
- Para la especificación de la métrica de escalado, la métrica que resulta de mayor utilidad para escalar es una métrica de rendimiento o uso promedio por instancia.

- La métrica de escalado debe ser inversamente proporcional a la capacidad. Es decir, si el número de instancias en el grupo de escalado automático aumenta, la métrica de escalado debería disminuir aproximadamente en la misma proporción. Para garantizar que el escalado predictivo se comporte según lo esperado, la métrica de carga y la métrica de escalado también deben estar estrechamente correlacionadas entre sí.
- El uso objetivo debe coincidir con el tipo de métrica de escalado. Para configurar una política que emplee la utilización de la CPU, se trata de un porcentaje objetivo. Para la configuración de una política que use el rendimiento, como el número de solicitudes o mensajes, este es el número objetivo de solicitudes o mensajes por instancia durante cualquier intervalo de un minuto.
- Si no se siguen estas recomendaciones, es probable que los valores futuros pronosticados de la serie temporal sean incorrectos. Para validar que los datos son correctos, puede ver los valores pronosticados en la consola de Amazon EC2 Auto Scaling. Como alternativa, después de crear la política de escalado predictivo, inspeccione los `CapacityForecast` objetos devueltos por una llamada a la GetPredictiveScalingForecastAPI `LoadForecast` y los objetos devueltos por ella.
- Se recomienda configurar el escalado predictivo en modo Forecast only (Solo pronóstico) para poder evaluar el pronóstico antes de que el escalado predictivo comience a escalar la capacidad de forma activa.

Requisitos previos

Para agregar métricas personalizadas en la política de escalado predictivo, debe tener permisos de `cloudwatch:GetMetricData`.

Para especificar sus propias métricas en lugar de las métricas que AWS proporciona, primero debe publicarlas en CloudWatch. Para obtener más información, consulta Publicar métricas personalizadas en la Guía del CloudWatch usuario de Amazon.

Si publica sus propias métricas, asegúrese de publicar los puntos de datos con una frecuencia mínima de cinco minutos. Amazon EC2 Auto Scaling recupera los puntos CloudWatch de datos en función de la duración del período que necesite. Por ejemplo, la especificación de métrica de carga utiliza métricas por hora para medir la carga de la aplicación. CloudWatch utiliza los datos de las métricas publicados para proporcionar un único valor de datos para cualquier período de una hora al agregar todos los puntos de datos con las marcas de tiempo correspondientes a cada período de una hora.

Construir JSON para métricas personalizadas

La siguiente sección contiene ejemplos sobre cómo configurar el escalado predictivo desde el que consultar datos. CloudWatch Existen dos métodos diferentes para configurar esta opción, y el método que elija afectará al formato que utilice para construir el JSON para su política de escalado predictivo. Cuando usa matemáticas métricas, el formato del JSON varía aún más en función de las matemáticas métricas que se estén desempeñando.

1. Para crear una política que obtenga datos directamente de otras CloudWatch métricas proporcionadas AWS o en las que publique CloudWatch, consulteEjemplo de política de escalado predictivo con métricas de carga y escalado personalizadas (AWS CLI).
2. Para crear una política que pueda consultar varias CloudWatch métricas y utilizar expresiones matemáticas para crear nuevas series temporales basadas en estas métricas, consulteUso de expresiones de cálculos de métricas.

Ejemplo de política de escalado predictivo con métricas de carga y escalado personalizadas (AWS CLI)

Para crear una política de escalado predictivo con métricas de carga y escalado personalizadas con AWS CLI, almacene los argumentos `--predictive-scaling-configuration` en un archivo JSON denominado`config.json`.

Para empezar a agregar métricas personalizadas, sustituya los valores reemplazables del siguiente ejemplo por los de sus métricas y su utilización objetivo.

```
{
  "MetricSpecifications": [
    {
      "TargetValue": 50,
      "CustomizedScalingMetricSpecification": {
        "MetricDataQueries": [
          {
            "Id": "scaling_metric",
            "MetricStat": {
              "Metric": {
                "MetricName": "MyUtilizationMetric",
                "Namespace": "MyNameSpace",
                "Dimensions": [
                  {
                    "Name": "MyOptionalMetricDimensionName",
```

```
                    "Value": "MyOptionalMetricDimensionValue"
                  }
                ]
              },
              "Stat": "Average"
            }
          }
        ]
      },
      "CustomizedLoadMetricSpecification": {
        "MetricDataQueries": [
          {
            "Id": "load_metric",
            "MetricStat": {
              "Metric": {
                "MetricName": "MyLoadMetric",
                "Namespace": "MyNameSpace",
                "Dimensions": [
                  {
                    "Name": "MyOptionalMetricDimensionName",
                    "Value": "MyOptionalMetricDimensionValue"
                  }
                ]
              },
              "Stat": "Sum"
            }
          }
        ]
      }
    }
  ]
}
```

Para obtener más información, consulte la MetricDataQueryreferencia de la API Amazon EC2 Auto Scaling.

> **Note**
>
> Los siguientes son algunos recursos adicionales que pueden ayudarle a encontrar nombres de métricas, espacios de nombres, dimensiones y estadísticas para las métricas: CloudWatch

- Para obtener información sobre las métricas disponibles para AWS los servicios, consulta AWS los servicios que publican CloudWatch métricas en la Guía del CloudWatch usuario de Amazon.
- Para obtener el nombre, el espacio de nombres y las dimensiones exactos (si corresponde) de una CloudWatch métrica con el AWS CLI, consulta list-metrics.

Para crear esta política, ejecute el comando put-scaling-policy con el archivo de JSON como entrada, tal y como se muestra en el ejemplo siguiente.

```
aws autoscaling put-scaling-policy --policy-name my-predictive-scaling-policy \
  --auto-scaling-group-name my-asg --policy-type PredictiveScaling \
  --predictive-scaling-configuration file://config.json
```

Si se ejecuta correctamente, este comando devuelve el nombre de recurso de Amazon (ARN) de la política.

```
{
  "PolicyARN": "arn:aws:autoscaling:region:account-id:scalingPolicy:2f4f5048-d8a8-4d14-
b13a-d1905620f345:autoScalingGroupName/my-asg:policyName/my-predictive-scaling-policy",
  "Alarms": []
}
```

Uso de expresiones de cálculos de métricas

En la siguiente sección se proporcionan información y ejemplos de políticas de escalado predictivo que muestran cómo puede utilizar las matemáticas métricas en su política.

Temas

- Descripción del cálculo de métricas
- Ejemplo de política de escalado predictivo que combina métricas mediante la matemática métrica (AWS CLI)
- Ejemplo de política de escalado predictivo para usar en un escenario de blue/green despliegue (AWS CLI)

Descripción del cálculo de métricas

Si lo único que quiere hacer es agregar los datos de las métricas existentes, las matemáticas CloudWatch métricas le ahorran el esfuerzo y el costo de publicar otra métrica en ella. CloudWatch Puede usar cualquier métrica que AWS proporcione y también puede usar las métricas que defina como parte de sus aplicaciones. Por ejemplo, es posible que desee calcular la cola de tareas pendientes de Amazon SQS por instancia. Para ello, se toma el número aproximado de mensajes disponibles que desea recuperar de la cola y se divide por la capacidad de funcionamiento del grupo de escalado automático.

Para obtener más información, consulta Uso de las matemáticas métricas en la Guía del CloudWatch usuario de Amazon.

Si decide utilizar una expresión de cálculo de métricas en su política de escalado predictivo, tenga en cuenta los siguientes aspectos:

- Las operaciones matemáticas métricas utilizan los puntos de datos de una combinación única de keys/value pares de métricas de nombre, espacio de nombres y dimensiones de la métrica.
- Puede usar cualquier operador aritmético (+ - */^), función estadística (como AVG o SUM) u otra función compatible. CloudWatch
- Puede utilizar tanto las métricas como los resultados de otras expresiones matemáticas en las fórmulas de la expresión matemática.
- Sus expresiones de cálculo de métricas se pueden formar con diferentes agrupaciones. Sin embargo, es una práctica recomendada para el resultado final de la agrupación que se utilice `Average` para la métrica de escalado y `Sum` para la métrica de carga.
- Todas las expresiones utilizadas en la especificación de una métrica deben devolver en última instancia una única serie temporal.

Para utilizar cálculos de métricas, haga lo siguiente:

- Elija una o más métricas. CloudWatch A continuación, cree la expresión. Para obtener más información, consulta Uso de las matemáticas métricas en la Guía del CloudWatch usuario de Amazon.
- Compruebe que la expresión matemática métrica es válida mediante la CloudWatch consola o la CloudWatch GetMetricDataAPI.

Ejemplo de política de escalado predictivo que combina métricas mediante la matemática métrica (AWS CLI)

En ocasiones, en lugar de especificar la métrica directamente, es posible que tenga que procesar primero sus datos de alguna manera. Por ejemplo, puede tener una aplicación que extraiga trabajo de una cola de Amazon SQS y puede querer utilizar el número de elementos en la cola como criterio para realizar un escalado predictivo. El número de mensajes en la cola no define exclusivamente el número de instancias que necesita. Por lo tanto, es necesario trabajar más para crear una métrica que pueda utilizarse para calcular las tareas pendientes por instancia. Para obtener más información, consulte Política de escalado basada en Amazon SQS.

A continuación se presenta un ejemplo de política de escalado predictivo para este caso. Especifica las métricas de escalado y carga que dependen de la métrica `ApproximateNumberOfMessagesVisible` de Amazon SQS, es decir, el número de mensajes disponibles para recuperar de la cola. Asimismo, utiliza la métrica `GroupInServiceInstances` de Amazon EC2 Auto Scaling y una expresión matemática que permite calcular las tareas pendientes por instancia para la métrica de escalado.

```
aws autoscaling put-scaling-policy --policy-name my-sqs-custom-metrics-policy \
  --auto-scaling-group-name my-asg --policy-type PredictiveScaling \
  --predictive-scaling-configuration file://config.json
{
  "MetricSpecifications": [
    {
      "TargetValue": 100,
      "CustomizedScalingMetricSpecification": {
        "MetricDataQueries": [
          {
            "Label": "Get the queue size (the number of messages waiting to be
 processed)",
            "Id": "queue_size",
            "MetricStat": {
              "Metric": {
                "MetricName": "ApproximateNumberOfMessagesVisible",
                "Namespace": "AWS/SQS",
                "Dimensions": [
                  {
                    "Name": "QueueName",
                    "Value": "my-queue"
                  }
                ]
```

```
            },
            "Stat": "Sum"
          },
          "ReturnData": false
        },
        {
          "Label": "Get the group size (the number of running instances)",
          "Id": "running_capacity",
          "MetricStat": {
            "Metric": {
              "MetricName": "GroupInServiceInstances",
              "Namespace": "AWS/AutoScaling",
              "Dimensions": [
                {
                  "Name": "AutoScalingGroupName",
                  "Value": "my-asg"
                }
              ]
            },
            "Stat": "Sum"
          },
          "ReturnData": false
        },
        {
          "Label": "Calculate the backlog per instance",
          "Id": "scaling_metric",
          "Expression": "queue_size / running_capacity",
          "ReturnData": true
        }
      ]
    },
    "CustomizedLoadMetricSpecification": {
      "MetricDataQueries": [
        {
          "Id": "load_metric",
          "MetricStat": {
            "Metric": {
              "MetricName": "ApproximateNumberOfMessagesVisible",
              "Namespace": "AWS/SQS",
              "Dimensions": [
                {
                  "Name": "QueueName",
                  "Value": "my-queue"
                }
```

```
                ],
              },
              "Stat": "Sum"
            },
            "ReturnData": true
          }
        ]
      }
    }
  ]
}
```

En el ejemplo se devuelve el ARN de la política.

```
{
  "PolicyARN": "arn:aws:autoscaling:region:account-id:scalingPolicy:2f4f5048-d8a8-4d14-
b13a-d1905620f345:autoScalingGroupName/my-asg:policyName/my-sqs-custom-metrics-policy",
  "Alarms": []
}
```

Ejemplo de política de escalado predictivo para usar en un escenario de blue/green despliegue (AWS CLI)

Una expresión de búsqueda proporciona una opción avanzada en la que se puede consultar una métrica de varios grupos de Auto Scaling y realizar expresiones matemáticas en ellos. Esto resulta especialmente útil para blue/green las implementaciones.

> **Note**
>
> Un blue/green despliegue es un método de despliegue en el que se crean dos grupos de Auto Scaling separados pero idénticos. Solo uno de ellos recibe el tráfico de producción. El tráfico de usuarios se dirige en principio al grupo de escalado automático (“azul”) anterior, mientras que un nuevo grupo (“verde”) se utiliza para probar y evaluar una nueva versión de una aplicación o de un servicio. El tráfico de usuarios pasa al grupo de escalado automático verde una vez que se prueba y acepta una nueva implementación. A continuación, puede eliminar el grupo azul una vez que la implementación se realiza correctamente.

Cuando se crean nuevos grupos de Auto Scaling como parte de una blue/green implementación, el historial de métricas de cada grupo se puede incluir automáticamente en la política de escalado

predictivo sin tener que cambiar sus especificaciones métricas. Para obtener más información, consulte Uso de las políticas de escalado predictivo de Auto Scaling de EC2 Blue/Green con implementaciones en AWS el blog de informática.

En el siguiente ejemplo de política se muestra cómo hacerlo. En este ejemplo, la política utiliza la métrica `CPUUtilization` emitida por Amazon EC2. Utiliza la métrica `GroupInServiceInstances` de Amazon EC2 Auto Scaling y una expresión matemática que permite calcular el valor de la métrica de escalado por instancia. También establece una especificación de la métrica de capacidad para obtener la métrica `GroupInServiceInstances`.

La expresión de búsqueda localiza la `CPUUtilization` de las instancias en varios grupos de Auto Scaling según los criterios de búsqueda especificados. Si posteriormente se crea un nuevo grupo de escalado automático que coincida con los mismos criterios de búsqueda, la `CPUUtilization` de las instancias del nuevo grupo se incluye de manera automática.

```
aws autoscaling put-scaling-policy --policy-name my-blue-green-predictive-scaling-
policy \
  --auto-scaling-group-name my-asg --policy-type PredictiveScaling \
  --predictive-scaling-configuration file://config.json
{
  "MetricSpecifications": [
    {
      "TargetValue": 25,
      "CustomizedScalingMetricSpecification": {
        "MetricDataQueries": [
          {
            "Id": "load_sum",
            "Expression": "SUM(SEARCH('{AWS/EC2,AutoScalingGroupName} MetricName=
\"CPUUtilization\" ASG-myapp', 'Sum', 300))",
            "ReturnData": false
          },
          {
            "Id": "capacity_sum",
            "Expression": "SUM(SEARCH('{AWS/AutoScaling,AutoScalingGroupName}
 MetricName=\"GroupInServiceInstances\" ASG-myapp', 'Average', 300))",
            "ReturnData": false
          },
          {
            "Id": "weighted_average",
            "Expression": "load_sum / capacity_sum",
            "ReturnData": true
          }
```

```
        ]
      },
      "CustomizedLoadMetricSpecification": {
        "MetricDataQueries": [
          {
            "Id": "load_sum",
            "Expression": "SUM(SEARCH('{AWS/EC2,AutoScalingGroupName} MetricName=
\"CPUUtilization\" ASG-myapp', 'Sum', 3600))"
          }
        ]
      },
      "CustomizedCapacityMetricSpecification": {
        "MetricDataQueries": [
          {
            "Id": "capacity_sum",
            "Expression": "SUM(SEARCH('{AWS/AutoScaling,AutoScalingGroupName}
 MetricName=\"GroupInServiceInstances\" ASG-myapp', 'Average', 300))"
          }
        ]
      }
    }
  ]
}
```

En el ejemplo se devuelve el ARN de la política.

```
{
  "PolicyARN": "arn:aws:autoscaling:region:account-id:scalingPolicy:2f4f5048-d8a8-4d14-
b13a-d1905620f345:autoScalingGroupName/my-asg:policyName/my-blue-green-predictive-
scaling-policy",
  "Alarms": []
}
```

Consideraciones sobre la utilización de métricas personalizadas en una política de escalado predictivo

Si se produce un problema durante el uso de las métricas personalizadas, se recomienda seguir los siguientes pasos:

- Si aparece un mensaje de error, léalo y solucione el problema que indica, en caso de que sea posible.

- Si se produce un problema al intentar utilizar una expresión de búsqueda en un escenario de blue/green despliegue, primero asegúrese de que sabe cómo crear una expresión de búsqueda que busque una coincidencia parcial en lugar de una coincidencia exacta. Además, compruebe que en su consulta solo se encuentren los grupos de Auto Scaling que ejecutan la aplicación específica. Para obtener más información sobre la sintaxis de las expresiones de búsqueda, consulte la sintaxis de las expresiones de CloudWatch búsqueda en la Guía del CloudWatch usuario de Amazon.
- Si no ha validado una expresión de antemano, el comando put-scaling-policy lo hace cuando crea la política de escalado. Sin embargo, existe la posibilidad de que este comando no identifique la causa exacta de los errores detectados. Para corregir esto, solucione los errores que recibe en la respuesta de una solicitud al comando get-metric-data. También puede solucionar los problemas de la expresión desde la CloudWatch consola.
- Cuando vea los gráficos de Load (Carga) y Capacity (Capacidad) en la consola, es posible que el gráfico de Capacity (Capacidad) no muestre ningún dato. Para garantizar que los gráficos tengan datos completos, asegúrese de habilitar de manera sistemática las métricas de grupo para sus grupos de Auto Scaling. Para obtener más información, consulte Habilitación de las métricas de grupo de Auto Scaling (consola).
- La especificación de métrica de capacidad solo es útil para blue/green las implementaciones cuando hay aplicaciones que se ejecutan en diferentes grupos de Auto Scaling a lo largo de su vida útil. Esta métrica personalizada le permite proporcionar la capacidad total de varios grupos de Auto Scaling. El escalado predictivo lo utiliza para mostrar datos históricos en los gráficos de Capacity (Capacidad) de la consola.
- Debe especificar `false` para `ReturnData` si `MetricDataQueries` especifica la función SEARCH() por sí sola sin una función matemática como SUM(). Esto se debe a que las expresiones de búsqueda podrían devolver varias series temporales, mientras que una especificación métrica basada en una expresión solo puede devolver una serie temporal.
- Todas las métricas que aparecen en una expresión de búsqueda deben tener la misma resolución.

Limitaciones

- Puede consultar puntos de datos de hasta 10 métricas en una especificación de métrica.
- A efectos de este límite, una expresión cuenta como una métrica.

Control de las instancias de Auto Scaling que se terminan durante una reducción horizontal

Amazon EC2 Auto Scaling utiliza políticas de terminación para decidir el orden de terminación de las instancias. Puede usar una política predefinida o crear una política personalizada para satisfacer sus requisitos específicos. Al utilizar una política personalizada o la protección frente a la reducción horizontal de instancias, también puede evitar que el grupo de escalado automático termine las instancias que aún no están listas para ser terminadas.

Contenido

- Cuándo se utilizan las políticas de terminación de Amazon EC2 Auto Scaling
- Configuración de políticas de terminación de Amazon EC2 Auto Scaling
- Creación de una política de terminación personalizada con Lambda
- Uso de la protección frente a la reducción horizontal de instancias para controlar la terminación de instancias
- Diseño de aplicaciones para gestionar sin problemas la terminación de instancias

Cuándo se utilizan las políticas de terminación de Amazon EC2 Auto Scaling

En las siguientes secciones se describen los escenarios en los que Amazon EC2 Auto Scaling utiliza políticas de terminación.

Contenido

- Eventos de reducción horizontal
- Actualización de instancias
- Reequilibrio de la zona de disponibilidad

Eventos de reducción horizontal

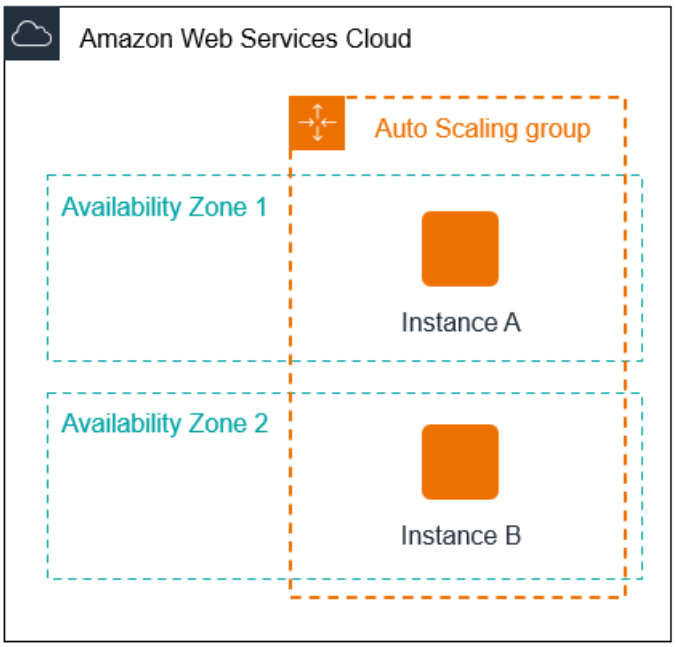
Un evento de reducción horizontal se produce cuando hay un nuevo valor para la capacidad deseada de un grupo de escalado automático que es inferior a la capacidad actual del grupo.

Los eventos de reducción horizontal ocurren en los siguientes escenarios:

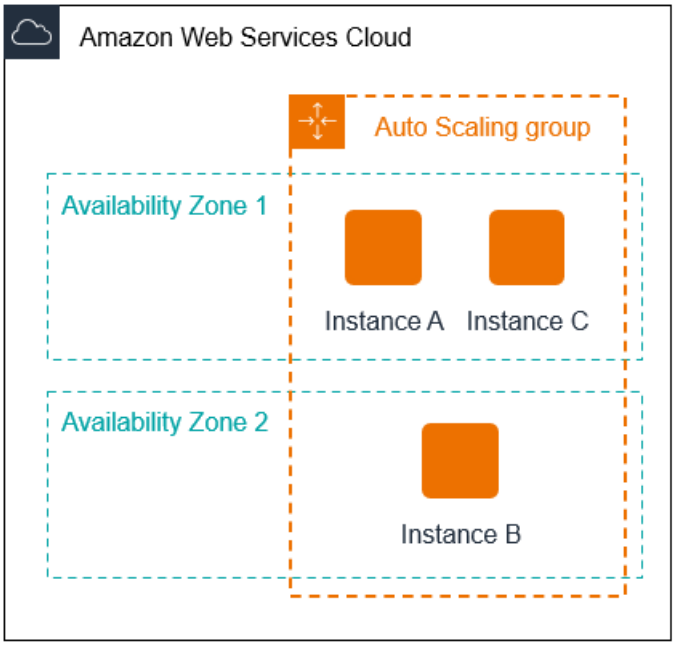
- Cuando se utilizan políticas de escalado dinámico y el tamaño del grupo disminuye como resultado de cambios en el valor de una métrica
- Cuando se utiliza el escalado programado y el tamaño del grupo disminuye como resultado de una acción programada
- Cuando reduce manualmente el tamaño del grupo

En el ejemplo siguiente se muestra cómo funcionan las políticas de terminación cuando hay un evento de reducción horizontal.

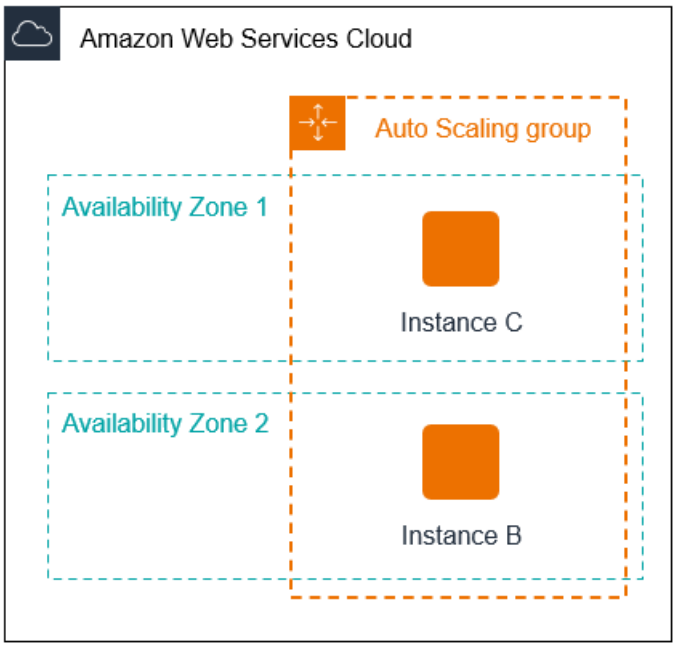
1. El grupo de escalado automático de este ejemplo tiene un tipo de instancia, dos zonas de disponibilidad y una capacidad deseada de dos instancias. También tiene una política de escalado dinámico que agrega y elimina instancias cuando la utilización de recursos aumenta o disminuye. Las dos instancias de este grupo se distribuyen entre las dos zonas de disponibilidad, como se muestra en el siguiente diagrama.



2. Cuando se reduce horizontalmente el grupo de escalado automático, Amazon EC2 Auto Scaling lanza una nueva instancia. El grupo de escalado automático ahora tiene tres instancias, distribuidas entre las dos zonas de disponibilidad, como se muestra en el siguiente diagrama.



3. Cuando se escala el grupo Auto Scaling, Amazon EC2 Auto Scaling termina una de las instancias.
4. Si no ha asignado una política de terminación específica al grupo, Amazon EC2 Auto Scaling utiliza la política de terminación predeterminada. Selecciona la zona de disponibilidad con dos instancias y termina la instancia lanzada desde una configuración de lanzamiento, una plantilla de lanzamiento distinta, o la versión más antigua de la plantilla de lanzamiento actual. Si las instancias se lanzaron desde la misma versión de la misma plantilla, Amazon EC2 Auto Scaling selecciona la instancia más próxima a la siguiente hora de facturación y la termina.



Actualización de instancias

Puede iniciar una actualización de instancias para actualizar las instancias del grupo de escalado automático. Durante una actualización de instancias, Amazon EC2 Auto Scaling termina las instancias del grupo y, a continuación, lanza reemplazos de las instancias terminadas. La política de terminación del grupo de escalado automático controla qué instancias se reemplazan primero.

Reequilibrio de la zona de disponibilidad

Amazon EC2 Auto Scaling equilibra la capacidad de manera uniforme en las zonas de disponibilidad habilitadas para el grupo de escalado automático. Esto ayuda a reducir el impacto de una interrupción en la zona de disponibilidad. Si la distribución de la capacidad entre las zonas de disponibilidad se desequilibra, Amazon EC2 Auto Scaling reequilibra el grupo de escalado automático lanzando instancias en las zonas de disponibilidad habilitadas con menos instancias y terminando instancias en otros lugares. La política de terminación controla qué instancias tienen prioridad para terminarlas primero.

Existen varias razones por las que la distribución de instancias entre las zonas de disponibilidad puede desequilibrarse.

Eliminación de instancias

Si desconecta instancias del grupo de escalado automático, pone instancias en espera o termina explícitamente instancias y disminuye la capacidad deseada, lo que impide que se lancen instancias de reemplazo, el grupo podría quedar desequilibrado. Si esto ocurre, Amazon EC2 Auto Scaling lo compensa reequilibrando las zonas de disponibilidad.

Uso de zonas de disponibilidad diferentes de las especificadas originalmente

Si expande el grupo de escalado automático para incluir zonas de disponibilidad adicionales, o cambia las zonas de disponibilidad que se utilizan, Amazon EC2 Auto Scaling lanzará instancias en las nuevas zonas de disponibilidad y las terminará en las otras zonas para garantizar que el grupo de escalado automático abarca de manera uniforme las zonas de disponibilidad.

Interrupción de disponibilidad

Las interrupciones de disponibilidad son raras. Sin embargo, si una zona de disponibilidad deja de estar disponible y se recupera posteriormente, el grupo de escalado automático puede quedar desequilibrado entre las zonas de disponibilidad. Amazon EC2 Auto Scaling intenta reequilibrar gradualmente el grupo y el reequilibrio puede terminar instancias en otras zonas.

Por ejemplo, imagine que hay un grupo de escalado automático que tiene un tipo de instancia, dos zonas de disponibilidad y una capacidad deseada de dos instancias. En una situación en la que se produce un error en una zona de disponibilidad, Amazon EC2 Auto Scaling lanza automáticamente una nueva instancia en la zona de disponibilidad en buen estado para reemplazar la de la zona de disponibilidad en mal estado. Cuando la zona de disponibilidad en mal estado vuelve a estar en buen estado, Amazon EC2 Auto Scaling lanza automáticamente una nueva instancia en esta zona, que a su vez termina una instancia en la zona no afectada.

> **Note**
>
> Durante el reequilibrio, Amazon EC2 Auto Scaling lanza nuevas instancias antes de terminar las antiguas, por lo que no se pone en peligro el rendimiento ni la disponibilidad de su aplicación.
>
> Como Amazon EC2 Auto Scaling intenta lanzar nuevas instancias antes de terminar las antiguas, cuando se está en la capacidad máxima especificada o cerca de ella puede impedir o detener completamente las actividades de reequilibrio. Para evitar este problema, el sistema puede superar temporalmente la capacidad máxima especificada de un grupo con un margen del 10 % (o con un margen de una instancia, lo que sea mayor) durante una actividad de reequilibrio. El margen solo se amplía si el grupo tiene o se aproxima a la capacidad máxima y necesita reequilibrarse, ya sea por una distribución de zonas solicitada por el usuario o para compensar los problemas de disponibilidad de zona. La extensión se dura solamente mientras sea necesaria para reequilibrar el grupo, normalmente unos minutos.

Configuración de políticas de terminación de Amazon EC2 Auto Scaling

Una política de terminación proporciona los criterios que Amazon EC2 Auto Scaling sigue para terminar las instancias en un orden específico. De forma predeterminada, Amazon EC2 Auto Scaling utiliza una política de terminación diseñada para terminar primero las instancias que utilizan configuraciones anticuadas. Puede cambiar la política de terminación para controlar qué instancias son más importantes cancelar primero.

Cuando Amazon EC2 Auto Scaling termina instancias, trata de mantener un equilibrio entre las zonas de disponibilidad habilitadas para el grupo de escalado automático. El mantenimiento del equilibrio entre las zonas tiene prioridad sobre las políticas de terminación. Si una zona de disponibilidad tiene más instancias que otras, Amazon EC2 Auto Scaling aplica primero la política

de terminación a la zona desequilibrada. Si las zonas de disponibilidad están equilibradas, aplica la política de terminación en todas las zonas.

Temas

- Funcionamiento de la política de terminación predeterminada
- Política de terminación predeterminada y grupos de instancias mixtas
- Políticas de terminación predefinidas
- Cambio de la política de terminación de un grupo de escalado automático

> **Note**
>
> Amazon EC2 Auto Scaling aplica políticas de terminación únicamente a las instancias que el grupo Auto Scaling no considere en mal estado. Como resultado, las instancias marcadas como insalubres por las comprobaciones de estado de Auto Scaling pasarán por alto la evaluación de la política de rescisión.
> Para obtener más información, consulte Diseño de aplicaciones para gestionar sin problemas la terminación de instancias.

Funcionamiento de la política de terminación predeterminada

Cuando Amazon EC2 Auto Scaling necesita terminar una instancia, primero determina qué zonas de disponibilidad tienen más instancias y busca al menos una instancia que no esté protegida frente a la reducción horizontal. A continuación, procede a evaluar las instancias desprotegidas dentro de la zona de disponibilidad identificada de la siguiente manera:

Instancias que utilizan configuraciones anticuadas

- Para los grupos que utilizan una plantilla de lanzamiento, determine si cualquiera de las instancias utiliza configuraciones obsoletas y establezca las prioridades en el siguiente orden:
  1. En primer lugar, compruebe si hay instancias lanzadas con una configuración de lanzamiento.
  2. Luego, compruebe si hay instancias lanzadas con una plantilla de lanzamiento diferente a la plantilla de lanzamiento actual.
  3. Por último, busque instancias que utilicen la versión más antigua de la plantilla de lanzamiento actual.

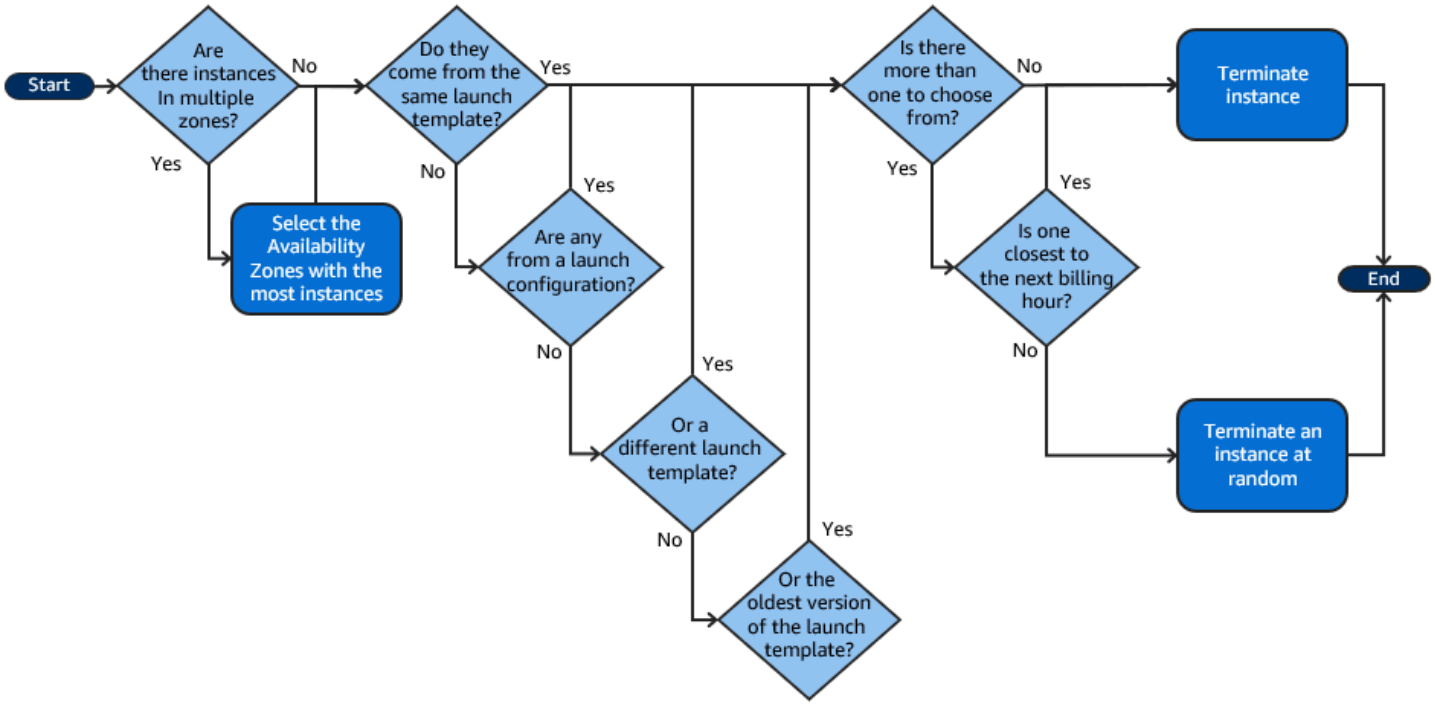
- Para los grupos que utilizan una configuración de lanzamiento, determine si alguna de estas instancias utiliza la configuración de lanzamiento más antigua.

Si no se encuentra ninguna instancia con configuraciones anticuadas o hay varias instancias entre las que elegir, Amazon EC2 Auto Scaling utiliza el siguiente criterio para las instancias que se acercan a su próxima hora de facturación.

Instancias cercanas a la próxima hora de facturación

Determine si cualquiera de las instancias que cumplen los criterios anteriores está más próxima que las demás a la siguiente hora de facturación. Si varias instancias están igual de próximas, cancele una al azar. Esto ayuda a maximizar el uso de las instancias que tienen un cargo por hora. Sin embargo, la mayor parte del uso de EC2 ahora se factura por segundo, por lo que esta optimización ofrece menos beneficios. Para obtener más información, consulte Precios de Amazon EC2.

El siguiente diagrama de flujo ilustra cómo funciona la política de terminación predeterminada para los grupos que utilizan una plantilla de lanzamiento.



Política de terminación predeterminada y grupos de instancias mixtas

Amazon EC2 Auto Scaling aplica criterios adicionales al finalizar instancias en grupos de instancias mixtos.

Cuando Amazon EC2 Auto Scaling necesita terminar una instancia, identifica primero qué opción de compra (spot o bajo demanda) debe ser terminada en función de la configuración del grupo. De este modo, se garantiza que el grupo tienda a alcanzar la proporción especificada de instancias de spot y bajo demanda a lo largo del tiempo.

A continuación, aplica la política de terminación en cada zona de disponibilidad individualmente. Determina qué instancia de spot o instancia bajo demanda se debe cerrar en qué zona de disponibilidad para mantener el equilibrio entre las zonas de disponibilidad. La misma lógica se aplica a los grupos de instancias mixtos que utilizan pesos definidos para los tipos de instancia.

Dentro de cada zona, la política de terminación predeterminada funciona de la siguiente manera para determinar qué instancia desprotegida que cumple con la opción de compra identificada puede terminarse:

1. Determine si alguna de las instancias se puede terminar para mejorar la alineación con la estrategia de asignación especificada para el grupo de escalado automático. Si no se identifica ninguna instancia para la optimización o hay varias instancias entre las que elegir, la evaluación continúa.
2. Determine si cualquiera de las instancias utiliza configuraciones obsoletas y establezca las prioridades en el siguiente orden:
   a. En primer lugar, compruebe si hay instancias lanzadas con una configuración de lanzamiento.
   b. Luego, compruebe si hay instancias lanzadas con una plantilla de lanzamiento diferente a la plantilla de lanzamiento actual.
   c. Por último, busque instancias que utilicen la versión más antigua de la plantilla de lanzamiento actual.

   Si no se encuentra ninguna instancia con configuraciones desactualizadas o hay varias instancias entre las que elegir, la evaluación continúa.
3. Determine si cualquiera de las instancias está más próxima que las demás a la siguiente hora de facturación. Si varias instancias están igual de próximas, elija una al azar.

Políticas de terminación predefinidas

Puede elegir entre las siguientes políticas de terminación predefinidas:

- **`Default`**: termina las instancias de acuerdo con la política de terminación predeterminada.

- **`AllocationStrategy`**: termina las instancias del grupo de escalado automático para alinear las instancias restantes con la estrategia de asignación para el tipo de instancias que se va a terminar (una instancia de spot o una instancia bajo demanda). Esta política es útil cuando han cambiado los tipos de instancias que prefiera. Si la estrategia de asignación de spot es `lowest-price`, puede reequilibrar gradualmente la distribución de instancias de spot en sus grupos de spot con los precios más bajos. Si la estrategia de asignación de spot es `capacity-optimized`, puede reequilibrar gradualmente la distribución de las instancias de spot en los grupos de spot donde hay más capacidad de spot disponibles. También puede reemplazar gradualmente las instancias bajo demanda de un tipo de prioridad menor por otras de un tipo de prioridad mayor.
- **`OldestLaunchTemplate`**: termina las instancias que tienen la plantilla de lanzamiento más antigua. Con esta política, las instancias que utilizan una plantilla de lanzamiento que no es la actual terminan primero, seguidas de las instancias que utilizan la versión más antigua de la plantilla de lanzamiento actual. Esta política es útil cuando va a actualizar un grupo y eliminar progresivamente las instancias de una configuración anterior.
- **`OldestLaunchConfiguration`**: termina las instancias que tienen la configuración de lanzamiento más antigua. Esta política es útil cuando va a actualizar un grupo y eliminar progresivamente las instancias de una configuración anterior. Con esta política, las instancias que usan la configuración de lanzamiento no actual se terminan primero.
- **`ClosestToNextInstanceHour`**: termina las instancias que están más próximas a la siguiente hora de facturación. Esta política ayuda a maximizar el uso de las instancias que tienen un cargo por hora.
- **`NewestInstance`**: termina la instancia más reciente del grupo. Esta política es útil cuando va a probar una nueva configuración de lanzamiento, pero no desea mantenerla en producción.
- **`OldestInstance`**: termina la instancia más antigua del grupo. Esta opción es útil cuando va a actualizar las instancias del grupo de escalado automático a un nuevo tipo de instancias EC2. Puede sustituir gradualmente las instancias del tipo antiguo por instancias del tipo nuevo.

> **Note**
>
> Amazon EC2 Auto Scaling siempre equilibra primero las instancias en las zonas de disponibilidad, independientemente de la política de terminación que se utilice. Como resultado, es posible que encuentre situaciones en las que algunas instancias más recientes se terminen antes de las instancias más antiguas. Por ejemplo, cuando hay una zona de disponibilidad agregada más recientemente, o cuando una zona de disponibilidad tiene más instancias que las otras zonas de disponibilidad usadas por el grupo.

Cambio de la política de terminación de un grupo de escalado automático

Para cambiar la política de terminación de su grupo de escalado automático, use uno de los siguientes métodos.

Console

No puede cambiar la política de terminación predeterminada cuando crea por primera vez un grupo de escalado automático en la consola de Amazon EC2 Auto Scaling. La política de terminación predeterminada se utiliza automáticamente. Una vez que cree el grupo de escalado automático, puede reemplazar la política predeterminada por una política de terminación diferente o por varias políticas de terminación enumeradas en el orden en el que deban aplicarse.

Cómo cambiar la política de terminación de un grupo de escalado automático

1. Abra la consola Amazon EC2 en https://console.aws.amazon.com/ec2/y seleccione Auto Scaling Groups en el panel de navegación.
2. Seleccione la casilla situada junto al grupo de escalado automático.

   Se abre un panel dividido en la parte inferior de la página.
3. En la pestaña Details (Detalles) elija (Advanced configurations) Configuraciones avanzadas, Edit (Editar).
4. Para Termination policies (Políticas de terminación), elija una o varias políticas de terminación. Si elige varias políticas, colóquelas en el orden en el que desea que se evalúen.

   De manera opcional, puede elegir Custom termination policy (Política de terminación personalizada) y luego elegir una función de Lambda que se ajuste a sus necesidades. Si ha creado versiones y alias para la función de Lambda, puede elegir una versión o un alias en el menú desplegable Version/Alias (Versión/Alias). Para utilizar la versión no publicada de la función de Lambda, mantenga Version/Alias (Versión/Alias) establecido en su valor predeterminado. Para obtener más información, consulte Creación de una política de terminación personalizada con Lambda.

   > **Note**
   >
   > Si se utilizan varias políticas, su orden debe establecerse correctamente:
   >
   > - Si utiliza la política Default (Predeterminada), debe ser la última de la lista.

- Si utiliza una política Custom termination (Terminación personalizada), debe ser la primera de la lista.

5. Elija Actualizar.

AWS CLI

La política de terminación predeterminada se utiliza automáticamente a menos que se especifique una política diferente.

Cómo cambiar la política de terminación de un grupo de escalado automático

Utilice uno de los siguientes comandos:

- create-auto-scaling-group
- update-auto-scaling-group

Puede utilizar políticas de terminación de manera individual o combinarlas en una lista de políticas. Por ejemplo, utilice el siguiente comando para actualizar un grupo de escalado automático para que utilice primero la política `OldestLaunchConfiguration` y después la política `ClosestToNextInstanceHour`.

```
aws autoscaling update-auto-scaling-group --auto-scaling-group-name my-asg --
termination-policies "OldestLaunchConfiguration" "ClosestToNextInstanceHour"
```

Si utiliza la política de terminación `Default`, colóquela la última en la lista de políticas de terminación. Por ejemplo, `--termination-policies "OldestLaunchConfiguration" "Default"`.

Para utilizar una política de rescisión personalizada, primero debe crear su política de rescisión utilizando AWS Lambda. Para especificar la función Lambda que se utilizará como política de terminación, colóquela la primera en la lista de políticas de terminación. Por ejemplo, `--termination-policies "arn:aws:lambda:us-west-2:123456789012:function:HelloFunction:prod" "OldestLaunchConfiguration"`. Para obtener más información, consulte Creación de una política de terminación personalizada con Lambda.

Creación de una política de terminación personalizada con Lambda

Amazon EC2 Auto Scaling utiliza políticas de terminación para priorizar qué instancias deben terminar primero al disminuir el tamaño del grupo de escalado automático (que se denomina reducción horizontal). El grupo de escalado automático utiliza una política de terminación predeterminada, pero opcionalmente puede elegir o crear sus propias políticas de terminación. Para obtener más información acerca de la elección de una política de terminación predefinida, consulte Configuración de políticas de terminación de Amazon EC2 Auto Scaling.

En este tema, aprenderá a crear una política de terminación personalizada utilizando una función AWS Lambda que Amazon EC2 Auto Scaling invoca en respuesta a determinados eventos. La función Lambda que crea procesa la información de los datos de entrada enviados por Amazon EC2 Auto Scaling y devuelve una lista de instancias listas para terminarse.

Una política de terminación personalizada proporciona un mejor control sobre qué instancias se terminan y cuándo. Por ejemplo, cuando el grupo de escalado automático se reduce horizontalmente, Amazon EC2 Auto Scaling no puede determinar si hay cargas de trabajo en ejecución que no deben interrumpirse. Con una función Lambda, puede validar la solicitud de terminación y esperar hasta que finalice la carga de trabajo antes de devolver el ID de instancia a Amazon EC2 Auto Scaling para la terminación.

Contenido

- Datos de entrada
- Datos de respuesta
- Consideraciones
- Crear la función de Lambda
- Limitaciones

Datos de entrada

Amazon EC2 Auto Scaling genera una carga útil JSON para los eventos de reducción horizontal, y también lo hace cuando las instancias están a punto de terminar como resultado de la duración máxima de la instancia o de las características de actualización de instancias. También genera una carga JSON para los eventos de reducción horizontal que puede iniciar al reequilibrar el grupo entre las zonas de disponibilidad.

Esta carga contiene información sobre la capacidad que Amazon EC2 Auto Scaling necesita terminar, una lista de instancias que sugiere para la terminación y el evento que inició la terminación.

A continuación, se muestra un ejemplo de carga:

```
{
  "AutoScalingGroupARN": "arn:aws:autoscaling:us-east-1:<account-
id>:autoScalingGroup:d4738357-2d40-4038-ae7e-b00ae0227003:autoScalingGroupName/my-asg",
  "AutoScalingGroupName": "my-asg",
  "CapacityToTerminate": [
    {
      "AvailabilityZone": "us-east-1b",
      "Capacity": 2,
      "InstanceMarketOption": "on-demand"
    },
    {
      "AvailabilityZone": "us-east-1b",
      "Capacity": 1,
      "InstanceMarketOption": "spot"
    },
    {
      "AvailabilityZone": "us-east-1c",
      "Capacity": 3,
      "InstanceMarketOption": "on-demand"
    }
  ],
  "Instances": [
    {
      "AvailabilityZone": "us-east-1b",
      "InstanceId": "i-0056faf8da3e1f75d",
      "InstanceType": "t2.nano",
      "InstanceMarketOption": "on-demand"
    },
    {
      "AvailabilityZone": "us-east-1c",
      "InstanceId": "i-02e1c69383a3ed501",
      "InstanceType": "t2.nano",
      "InstanceMarketOption": "on-demand"
    },
    {
      "AvailabilityZone": "us-east-1c",
      "InstanceId": "i-036bc44b6092c01c7",
      "InstanceType": "t2.nano",
      "InstanceMarketOption": "on-demand"
```

```
    },
    ...
  ],
  "Cause": "SCALE_IN"
}
```

La carga útil incluye el nombre del grupo de escalado automático, su Nombre de recurso de Amazon (ARN) y los siguientes elementos:

- `CapacityToTerminate` describe cuánta capacidad de spot o en diferido se ha configurado para terminar en una zona de disponibilidad determinada.
- `Instances` representa las instancias que Amazon EC2 Auto Scaling sugiere para la terminación en función de la información de `CapacityToTerminate`.
- `Cause` describe el evento que causó la terminación: `SCALE_IN`, `INSTANCE_REFRESH`, `MAX_INSTANCE_LIFETIME` o `REBALANCE`.

La siguiente información describe los factores más significativos de cómo Amazon EC2 Auto Scaling genera `Instances` en los datos de entrada:

- El mantenimiento del equilibrio entre las zonas de disponibilidad tiene prioridad cuando se termina una instancia debido a eventos de reducción horizontal y terminaciones basadas en la actualización de instancias. Por lo tanto, si una zona de disponibilidad tiene más instancias que las otras zonas de disponibilidad usadas por el grupo, los datos de entrada contienen instancias que se pueden elegir para la terminación solo de la zona de disponibilidad desequilibrada. Si las zonas de disponibilidad utilizadas por el grupo están equilibradas, los datos de entrada contienen instancias de todas las zonas de disponibilidad del grupo.
- Cuando se utiliza una política de instancias mixtas, mantener las capacidades de spot y en diferido en equilibrio en función de los porcentajes deseados para cada opción de compra también tiene prioridad. En primer lugar, identificamos cuál de los dos tipos (spot o en diferido) debe terminarse. A continuación, identificamos qué instancias (dentro de la opción de compra identificada) de qué zonas de disponibilidad podemos terminar, lo que hará que las zonas de disponibilidad estén más equilibradas.

Datos de respuesta

Los datos de entrada y los de respuesta en combinación reducen la lista de instancias que se van a terminar.

Con la entrada dada, la respuesta de la función Lambda debería ser similar a la del siguiente ejemplo:

```
{
  "InstanceIDs": [
    "i-02e1c69383a3ed501",
    "i-036bc44b6092c01c7",
    ...
  ]
}
```

El elemento `InstanceIDs` de la respuesta representa las instancias que están listas para terminar.

Como alternativa, puede devolver un conjunto diferente de instancias que están listas para terminarse, que reemplaza a las instancias en los datos de entrada. Si no hay instancias listas para terminar cuando se invoca la función Lambda, también puede optar por no devolver ninguna instancia.

Cuando no haya ninguna instancia lista para terminar, la respuesta de la función Lambda debería ser similar a la del siguiente ejemplo:

```
{
  "InstanceIDs": [ ]
}
```

Consideraciones

Tenga en cuenta las siguientes consideraciones al utilizar una política de terminación personalizada:

- Devolver una instancia primero en los datos de respuesta no garantiza su terminación. Si se devuelve un número de instancias superior al requerido cuando se invoca la función Lambda, Amazon EC2 Auto Scaling evalúa cada instancia con respecto a las demás políticas de terminación especificadas para el grupo de escalado automático. Cuando hay varias políticas de terminación, intenta aplicar la siguiente política de terminación de la lista y, si hay más instancias de las necesarias para terminar, pasa a la siguiente política de terminación, etc. Si no se especifica ninguna otra política de terminación, se utiliza la política de terminación predeterminada para determinar qué instancias se van a terminar.
- Si no se devuelven instancias o se agota el tiempo de espera de la función Lambda, Amazon EC2 Auto Scaling espera un breve periodo de tiempo antes de volver a invocar la función. Para cualquier evento de reducción horizontal, sigue intentándolo siempre y cuando la capacidad

deseada del grupo sea menor que su capacidad actual. Por ejemplo, en terminaciones basadas en actualizaciones, sigue intentándolo durante una hora. Después de eso, si sigue sin poder terminar instancias, se produce un error en la operación de actualización de instancias. Con la vida útil máxima de la instancia, Amazon EC2 Auto Scaling sigue intentando terminar la instancia que se ha identificado que supera su vida útil máxima.

- Debido a que la función se reintenta repetidamente, asegúrese de probar y corregir cualquier error permanente en el código antes de usar una función Lambda como una política de terminación personalizada.
- Si reemplaza los datos de entrada por su propia lista de instancias que desea terminar y al terminar estas instancias se desequilibran las zonas de disponibilidad, Amazon EC2 Auto Scaling reequilibra gradualmente la distribución de la capacidad entre las zonas de disponibilidad. Primero, invoca la función Lambda para ver si hay instancias que están listas para terminarse y poder determinar si debe comenzar a reequilibrar. Si hay instancias listas para terminarse, primero lanza nuevas instancias. Cuando las instancias terminan de lanzarse, detecta que la capacidad actual del grupo es superior a la capacidad deseada e inicia un evento de reducción horizontal.
- Una política de terminación personalizada no afecta a su capacidad de utilizar también la protección contra la reducción horizontal para evitar que determinadas instancias sean terminadas. Para obtener más información, consulte Uso de la protección frente a la reducción horizontal de instancias para controlar la terminación de instancias.
- Si la función Lambda no devuelve ninguna instancia, esto no impide automáticamente todas las terminaciones. Amazon EC2 Auto Scaling seguirá cancelando las instancias que las comprobaciones de estado de Auto Scaling consideren en mal estado, independientemente de las políticas de terminación.

Crear la función de Lambda

Comience por crear la función Lambda, de modo que pueda especificar su Nombre de recurso de Amazon (ARN) en las políticas de terminación del grupo de escalado automático.

Para crear una función Lambda (consola)

1. Abra la página de Funciones en la consola de Lambda.
2. En la barra de navegación de la parte superior de la pantalla, seleccione la misma región que utilizó cuando creó el grupo de escalado automático.
3. Elija Create function (Crear función) y, a continuación, elija Author from scratch (Crear desde cero).

4. En Basic information (Información básica), para Function name (Nombre de función), escriba un nombre para la función.
5. Seleccione Creación de función. Volverá al código y la configuración de la función.
6. Con la función aún abierta en la consola, en Function code (Código de función), pegue el código en el editor.
7. Elija Implementar.
8. Opcionalmente, cree una versión publicada de la función Lambda eligiendo la pestaña Versions (Versiones) y, a continuación, Publish new version (Publicar nueva versión). Para obtener más información acerca del control de versiones en Lambda, consulte Versiones de la función de Lambda en la Guía para desarrolladores de AWS Lambda .
9. Si eligió publicar una versión, elija la pestaña Aliases (Alias) si desea asociar un alias a esta versión de la función Lambda. Para obtener más información acerca de los alias en Lambda, consulte Alias de función Lambda en la Guía para desarrolladores de AWS Lambda
10. A continuación, elija la pestaña Configuration (Configuración) y luego Permissions (Permisos).
11. Desplácese hasta Política basada en recursos y elija Agregar permisos. Una política basada en recursos se utiliza para conceder permisos para invocar la función a la entidad principal que se especifica en la política. En este caso, la entidad principal será el rol vinculado al servicio de Amazon EC2 Auto Scaling asociado al grupo de escalado automático.
12. En la sección Policy statement (Instrucción de la política), configure sus permisos:
    a. Elija Cuenta de AWS.
    b. En Principal (Entidad principal), escriba el ARN del rol vinculado al servicio de llamada, por ejemplo, **`arn:aws:iam::<aws-account-id>:role/aws-service-role/autoscaling.amazonaws.com/AWSServiceRoleForAutoScaling`**.
    c. En Acción, elija lambda:. InvokeFunction
    d. En Statement ID (ID de estado), escriba un ID de instrucción único, como **`AllowInvokeByAutoScaling`**.
    e. Seleccione Save.
13. Después de seguir estas instrucciones, continúe para especificar el ARN de la función en las políticas de terminación para el grupo de escalado automático. Para obtener más información, consulte Cambio de la política de terminación de un grupo de escalado automático.

> **Note**
>
> Para ver ejemplos que puede utilizar como referencia para desarrollar su función Lambda, consulte el GitHub repositorio de Amazon EC2 Auto Scaling.

Limitaciones

- Solo puede especificar una función Lambda en las políticas de terminación para un grupo de escalado automático. Si se especifican varias políticas de terminación, primero se debe especificar la función Lambda.
- Puede hacer referencia a la función Lambda utilizando un ARN no calificado (sin sufijo) o un ARN calificado que tenga una versión o un alias como sufijo. Si se utiliza un ARN no calificado (por ejemplo, `function:my-function`), la política basada en recursos debe crearse en la versión no publicada de la función. Si se utiliza un ARN calificado (por ejemplo, `function:my-function:1` o `function:my-function:prod`), la política basada en recursos debe crearse en esa versión de la función publicada específica.
- No puede utilizar un ARN calificado con el sufijo `$LATEST`. Si intenta agregar una política de terminación personalizada que haga referencia a un ARN calificado con el sufijo `$LATEST`, se producirá un error.
- El número de instancias proporcionadas en los datos de entrada está limitado a 30 000 instancias. Si hay más de 30 000 instancias que podrían terminarse, los datos de entrada incluyen `"HasMoreInstances": true` para indicar que se devuelve el número máximo de instancias.
- El tiempo máximo de ejecución de la función Lambda es de dos segundos (2000 milisegundos). Como práctica recomendada, debe establecer el valor de tiempo de espera de la función Lambda según el tiempo de ejecución previsto. Las funciones Lambda tienen un tiempo de espera predeterminado de tres segundos, pero esto puede reducirse.
- Si su tiempo de ejecución supera el límite de 2 segundos, cualquier acción de reducción horizontal se suspenderá hasta que el tiempo de ejecución caiga por debajo de este umbral. En el caso de las funciones de Lambda con tiempos de ejecución más largos y constantes, busque una forma de reducir el tiempo de ejecución, por ejemplo, almacenando en caché los resultados donde puedan recuperarse durante las siguientes invocaciones a Lambda.
- No se pueden usar funciones de Lambda de cuentas diferentes. La función Lambda utilizada como política de terminación personalizada debe estar en el mismo Cuenta de AWS grupo que el de Auto Scaling. Las instrucciones de configuración de esta página sirven específicamente para configuraciones en una misma cuenta.

- Las políticas de terminación personalizadas no se aplican a las instancias en mal estado. Las instancias en mal estado siempre se consideran aptas para la terminación, y la función Lambda solo evalúa la orden de terminación de las instancias restantes.

Uso de la protección frente a la reducción horizontal de instancias para controlar la terminación de instancias

La protección contra la reducción horizontal en instancias le permite controlar qué instancias de Amazon EC2 Auto Scaling se pueden terminar. Un caso de uso común para esta característica es escalar las cargas de trabajo basadas en contenedores. Para obtener más información, consulte Diseño de aplicaciones para gestionar sin problemas la terminación de instancias.

De forma predeterminada, la protección frente a la reducción horizontal de instancias está deshabilitada cuando se crea un grupo de escalado automático. Esto significa que Amazon EC2 Auto Scaling puede terminar cualquier instancia del grupo.

Puede proteger las instancias en cuanto se lanzan habilitando la configuración de protección frente a la reducción horizontal de instancias en el grupo de escalado automático. La protección de reducción horizontal de instancias comienza cuando la instancia tiene el estado `InService`. A continuación, para controlar qué instancias pueden terminar, deshabilite la configuración de protección contra la reducción horizontal en las instancias individuales del grupo de escalado automático. De este modo, puede seguir protegiendo determinadas instancias de las terminaciones no deseadas.

Temas

- Consideraciones
- Cambio en la protección contra la reducción horizontal para un grupo de escalado automático
- Cambio en la protección frente a la reducción horizontal de una instancia

Consideraciones

A continuación, se muestran las consideraciones que se deben tener en cuenta cuando se utiliza la protección frente a la reducción horizontal de instancias:

- Si todas las instancias de un grupo de escalado automático están protegidas frente a la reducción horizontal y se produce un evento de reducción horizontal, se reduce la capacidad deseada. Sin embargo, el grupo de escalado automático no puede terminar el número necesario de

instancias hasta que se desactiva la configuración de protección frente a la reducción horizontal de instancias. En el Consola de administración de AWS, el historial de actividades del grupo Auto Scaling incluye el siguiente mensaje si todas las instancias de un grupo de Auto Scaling están protegidas de la escalabilidad cuando se produce un evento de escalado: `Could not scale to desired capacity because all remaining instances are protected from scale in.`

- Si desconecta una instancia que está protegida frente a la reducción horizontal, se pierde la configuración de protección frente a la reducción horizontal de instancias. Cuando vuelve a asociar la instancia al grupo, esta hereda la configuración de protección de reducción horizontal de instancias actual del grupo. Cuando Amazon EC2 Auto Scaling lanza una nueva instancia o traslada una instancia de un grupo activo al grupo de escalado automático, esta hereda la configuración de protección frente a la reducción horizontal de instancias del grupo de escalado automático.
- La protección frente a la reducción horizontal de instancias no protege las instancias de Auto Scaling de lo siguiente:
  - La sustitución de comprobaciones de estado si la instancia no supera las comprobaciones de estado. Para obtener más información, consulte Comprobaciones de estado para instancias en un grupo de escalado automático.
  - Interrupciones de instancias de spot Las instancias de spot se terminan cuando la capacidad ya no está disponible o cuando el precio de spot supera el precio máximo.
  - Termina una reserva de bloques de capacidad. Amazon EC2 recupera las instancias del bloque de capacidad incluso si están protegidas contra la reducción horizontal.
  - Terminación manual mediante el comando de `terminate-instance-in-auto-scaling-group`. Para obtener más información, consulte Finalización de una instancia en su grupo de escalado automático (AWS CLI).
  - La terminación manual mediante la consola de Amazon EC2, los comandos de la CLI y las operaciones de API. Para proteger las instancias de Auto Scaling frente a la terminación manual, habilite la protección frente a la terminación de Amazon EC2. (Esto no impide que Amazon EC2 Auto Scaling termine las instancias o la terminación manual mediante el comando `terminate-instance-in-auto-scaling-group`). Para obtener información sobre cómo habilitar la protección contra la terminación de Amazon EC2 en una plantilla de lanzamiento, consulte Creación de una plantilla de lanzamiento mediante la configuración avanzada.

Cambio en la protección contra la reducción horizontal para un grupo de escalado automático

Puede habilitar o desactivar la configuración de protección frente a la reducción horizontal de instancias para un grupo de escalado automático. Al habilitarla, todas las instancias nuevas que lance el grupo tendrán habilitada la protección contra la reducción horizontal de instancias.

La activación o desactivación de esta configuración para un grupo de escalado automático no afecta a las instancias existentes.

Console

Cómo habilitar la protección contra la reducción horizontal para un grupo de escalado automático nuevo

Al crear el grupo de escalado automático, en la página Configurar políticas de escalado y tamaño de grupo, en Protección de reducción horizontal de instancias, seleccione la casilla Habilitar la protección contra la reducción horizontal de instancias.

Cómo habilitar o deshabilitar la protección contra la reducción horizontal para un grupo existente

1. Abra la consola Amazon EC2 en https://console.aws.amazon.com/ec2/y seleccione Auto Scaling Groups en el panel de navegación.
2. Seleccione la casilla situada junto al grupo de escalado automático.

   Se abre un panel dividido en la parte inferior de la página.
3. En la pestaña Details (Detalles) elija (Advanced configurations) Configuraciones avanzadas, Edit (Editar).
4. Para la protección contra la reducción horizontal de instancias, active o desactive la casilla Habilitar la protección contra la reducción horizontal de instancias para activar o desactivar esta opción según sea necesario.
5. Elija Actualizar.

AWS CLI

Cómo habilitar la protección contra la reducción horizontal para un grupo de escalado automático nuevo

Utilice el siguiente comando create-auto-scaling-group para habilitar la protección de escalado descendente de instancias.

```
aws autoscaling create-auto-scaling-group --auto-scaling-group-name my-asg --new-instances-protected-from-scale-in ...
```

Cómo habilitar la protección contra la reducción horizontal para un grupo existente

Usa el siguiente update-auto-scaling-groupcomando para habilitar la protección de escalado interno de instancias para el grupo de Auto Scaling especificado.

```
aws autoscaling update-auto-scaling-group --auto-scaling-group-name my-asg --new-instances-protected-from-scale-in
```

Cómo deshabilitar la protección contra la reducción horizontal para un grupo existente

Utilice el siguiente comando para desactivar la protección de reducción horizontal de instancias para el grupo especificado.

```
aws autoscaling update-auto-scaling-group --auto-scaling-group-name my-asg --no-new-instances-protected-from-scale-in
```

Cambio en la protección frente a la reducción horizontal de una instancia

De forma predeterminada, una instancia obtiene la configuración de protección frente a la reducción horizontal de instancias de su grupo de escalado automático. Sin embargo, puede habilitar o deshabilitar la protección contra la reducción horizontal de instancias para instancias individuales luego de lanzarlas.

Console

Cómo habilitar o deshabilitar la protección contra la reducción horizontal para una instancia

1. Abra la consola Amazon EC2 en https://console.aws.amazon.com/ec2/y seleccione Auto Scaling Groups en el panel de navegación.
2. Seleccione la casilla situada junto al grupo de escalado automático.

   Se abre un panel dividido en la parte inferior de la página.

3. En la pestaña Instance management (Administración de instancias), en Instances (Instancias), seleccione una instancia.
4. Para habilitar la protección de reducción horizontal de instancias, elija Acciones, Establecer protección de reducción horizontal. Cuando se lo pidan, seleccione Establecer protección de reducción horizontal.
5. Para deshabilitar la protección de reducción horizontal de instancias, seleccione Acciones, Eliminar protección de reducción horizontal. Cuando se lo pidan, seleccione Eliminar protección de reducción horizontal.

AWS CLI

Cómo habilitar la protección contra la reducción horizontal para una instancia

Utilice el siguiente comando set-instance-protection para habilitar la protección de escalado descendente de instancias para la instancia especificada.

```
aws autoscaling set-instance-protection --instance-ids i-5f2e8a0d --auto-scaling-
group-name my-asg --protected-from-scale-in
```

Cómo deshabilitar la protección contra la reducción horizontal para una instancia

Utilice el siguiente comando para desactivar la protección de reducción horizontal de instancias para la instancia especificada.

```
aws autoscaling set-instance-protection --instance-ids i-5f2e8a0d --auto-scaling-
group-name my-asg --no-protected-from-scale-in
```

> **Note**
>
> Recuerde que la protección frente a la reducción horizontal de instancias no garantiza que las instancias no se terminen en caso de error humano; por ejemplo, si alguien termina manualmente una instancia mediante la consola de Amazon EC2 o la AWS CLI. Para proteger la instancia de una terminación accidental, puede utilizar la protección frente a la terminación de Amazon EC2. Sin embargo, incluso con la protección frente a la terminación y la protección frente a la reducción horizontal de instancias habilitadas, los datos guardados en el almacenamiento de instancias pueden perderse si una comprobación de estado determina que una instancia no está en buen estado o si el grupo se elimina

accidentalmente. Al igual que en cualquier entorno, una práctica recomendada es realizar copias de seguridad de sus datos con frecuencia o cuando sea apropiado para los requisitos de continuidad de la empresa.

Diseño de aplicaciones para gestionar sin problemas la terminación de instancias

En este tema se describen las funciones que puede utilizar para impedir que su grupo de Amazon EC2 Auto Scaling cancele las instancias de Amazon EC2 que aún no están listas para finalizar. De forma predeterminada, Auto Scaling no tiene visibilidad de las aplicaciones que se ejecutan en sus instancias. Puede finalizar las instancias antes de que la aplicación pueda cerrar o completar las tareas asignadas sin problemas. Estas funciones le dan tiempo a la aplicación para completar el trabajo en curso, transferir el estado o realizar una limpieza antes de que finalice la instancia. Puede utilizarlas individualmente o en combinación, en función de los requisitos de la aplicación.

Estas funciones son especialmente útiles para las cargas de trabajo con estado, en las que cada instancia de la flota contiene datos, trabajos o estados diferentes a los de otras instancias. La finalización de las instancias activas sin un cierre correcto puede provocar que los trabajos que se estén ejecutando durante mucho tiempo se reinicien desde el principio, se reduzca la redundancia o la pérdida de datos y se interrumpan las transacciones o los cálculos en curso. Para cerrar correctamente una instancia con estado, su carga de trabajo debe agotarse (completando todos los trabajos actualmente asignados) o transferirse (mover los trabajos, los datos o la configuración a otra instancia activa).

Contenido

- Enlaces de ciclo de vida de terminación
- Protección contra la reducción horizontal de instancias
- Política de terminación personalizada
- Política del ciclo de vida de las
- Suspenda las terminaciones por completo
- Limitaciones
- Ejemplos de escenarios de

Enlaces de ciclo de vida de terminación

Un enlace del ciclo de vida de terminación prolonga la vida útil de la instancia de Amazon EC2 que ya está seleccionada para la terminación. Proporciona tiempo adicional para completar el trabajo en curso asignado actualmente a la instancia o para guardar el progreso y transferir el trabajo a otra instancia.

En el caso de muchas cargas de trabajo, una interrupción del ciclo de vida de terminación puede ser suficiente para cerrar sin problemas una aplicación en una instancia seleccionada para su finalización. Este es el enfoque que se hace todo lo posible y no se puede utilizar para impedir la rescisión si se abandona una acción relacionada con el ciclo de vida de la rescisión. Si su carga de trabajo tiene una baja tolerancia a los fallos en las acciones del ciclo de vida de las instancias, configure las políticas del ciclo de vida de las instancias en combinación con los enlaces del ciclo de vida de las terminaciones para conservar las instancias.

Para usar un enlace del ciclo de vida de las terminaciones, debes saber cuándo se selecciona una instancia para su terminación. Tiene dos formas de saberlo:

Opción	Description (Descripción)	Más adecuado para	Enlace a la documentación
Dentro de la instancia	El Servicio de metadatos de instancias (IMDS) es un punto de conexión seguro en el que se puede sondear el estado de una instancia directamente desde la instancia. Si los metadatos arrojan `Terminated` , entonces está previsto que la instancia sea terminada.	Aplicaciones en las que debe realizar una acción en la instancia antes de que se cierre.	Recupera el estado del ciclo de vida objetivo
Fuera de la instancia	Cuando una instancia finaliza, se genera una notificación de evento. Puede crear reglas con Amazon EventBridge, Amazon SQS, Amazon SNS	Aplicaciones que deben realizar acciones fuera de la instancia.	Configure un objetivo de notificación

Opción	Description (Descripción)	Más adecuado para	Enlace a la documentación
	AWS Lambda o para capturar estos eventos e invocar una respuesta, por ejemplo, con una función Lambda.		

Para usar un enlace de ciclo de vida, también necesitas saber cuándo la instancia está lista para finalizar por completo. Amazon EC2 Auto Scaling no finalizará la instancia hasta que reciba CompleteLifecycleActionuna llamada a la API o hasta que se agote el tiempo de espera, lo que ocurra primero.

De forma predeterminada, una instancia puede seguir ejecutándose durante una hora (tiempo de espera) debido a un enlace de ciclo de vida de terminación. Puede configurar el tiempo de espera predeterminado si una hora no es suficiente para completar la acción del ciclo de vida. Cuando una acción del ciclo de vida esté en curso, puede extender el tiempo de espera con llamadas a la API. RecordLifecycleActionHeartbeat

Para obtener más información, consulte Enlaces de ciclo de vida de Amazon EC2 Auto Scaling.

Protección contra la reducción horizontal de instancias

Puedes usar la protección escalable de instancias para controlar qué instancias se seleccionan para su cierre durante los eventos de escalamiento interno, especialmente para evitar que se termine una instancia que está procesando activamente un trabajo de larga duración. Por ejemplo, cuando se ejecutan cargas de trabajo en contenedores, es habitual querer proteger todas las instancias y eliminar la protección solo para las instancias que no tienen tareas actuales o programadas. Las instancias pueden seguir buscando nuevos trabajos y volver a habilitar la protección cuando se les asignen nuevos trabajos.

Puede habilitar la protección escalable a nivel de grupo e instancia de Auto Scaling. Al habilitar la protección escalable en el nivel de grupo de Auto Scaling, solo se protegen las instancias nuevas cuando se crean. En el caso de las instancias existentes, puede habilitar la protección de forma individual.

Las aplicaciones pueden configurar la protección desde las propias instancias o desde un plano de control centralizado que gestiona si cada instancia es finalizable. Recomendamos el enfoque

centralizado para flotas grandes o cuando sea necesario cambiar la protección con frecuencia, ya que permite realizar llamadas por lotes SetInstanceProtectiony evitar problemas de limitación de la API.

Para obtener más información, consulte Uso de la protección frente a la reducción horizontal de instancias para controlar la terminación de instancias.

Política de terminación personalizada

Al igual que la protección escalable de instancias, una política de terminación personalizada le ayuda a evitar que su grupo de Amazon EC2 Auto Scaling termine instancias EC2 específicas. Las instancias que no estén en buen estado pueden cancelarse independientemente de su política de cancelación personalizada.

Su grupo de Auto Scaling usa una política de terminación predeterminada para determinar qué instancias de Amazon EC2 termina primero. Si quieres tener más control sobre qué instancias terminan primero, puedes implementar una política de terminación personalizada mediante una función Lambda. Auto Scaling llama a esta función siempre que necesite seleccionar una instancia para su terminación y solo terminará las instancias que devuelva la función. Si la función produce un error, se agota el tiempo de espera o devuelve una lista vacía, Auto Scaling no termina ninguna instancia a menos que la instancia no esté en buen estado.

Una política de terminación personalizada resulta útil cuando la aplicación puede identificar qué instancias están inactivas o qué instancias se pueden terminar de forma segura. Por lo general, esto requiere un plano de control que haga un seguimiento de la carga de trabajo en todo el grupo.

Para obtener más información, consulte Creación de una política de terminación personalizada con Lambda.

Política del ciclo de vida de las

Las políticas del ciclo de vida de las instancias brindan protección contra las terminaciones de Amazon EC2 Auto Scaling cuando se abandona una acción del ciclo de vida de la terminación. A diferencia de los enlaces del ciclo de vida por sí solos, las políticas del ciclo de vida de las instancias están diseñadas para garantizar que las instancias pasen a un estado retenido cuando los procedimientos de cierre correctos no se completen correctamente.

Cuando Auto Scaling selecciona una instancia para su finalización, se invocan los enlaces del ciclo de vida de terminación configurados y la aplicación inicia procedimientos de cierre sin problemas.

Si las acciones del ciclo de vida de terminación se completan correctamente`CONTINUE`, la instancia finaliza normalmente. Sin embargo, si se abandona una acción de cancelación del ciclo de vida por cualquier motivo, la política del ciclo de vida de la instancia pasa la instancia a un estado retenido en lugar de finalizarla. Las instancias retenidas no cuentan para la capacidad deseada de su grupo de Auto Scaling, por lo que las instancias de reemplazo se lanzan automáticamente. Se le cobrarán los cargos estándar de Amazon EC2 tanto por la instancia retenida como por su reemplazo hasta que cancele manualmente la instancia retenida mediante TerminateInstanceInAutoScalingGroupla API.

Para usar esta función, debe configurar una política de ciclo de vida de la instancia con el activador de `TerminateHookAbandon` retención establecido en`retain`, así como al menos un enlace del ciclo de vida de la terminación. Dado que las instancias retenidas conllevan costos continuos de Amazon EC2 y requieren una acción manual, la supervisión es fundamental. Debería habilitar CloudWatch métricas como estas `GroupTerminatingRetainedInstances` y crear CloudWatch alarmas que le avisen cuando las instancias entren en estados de retención.

Para obtener más información, consulte Controle la retención de instancias con políticas de ciclo de vida de las instancias.

Suspenda las terminaciones por completo

Si necesita un control total sobre todas las terminaciones de instancias EC2 de su grupo Amazon EC2 Auto Scaling, suspenda el proceso. `Terminate` Solo recomendamos usar esta opción si las opciones anteriores no le ofrecen el control que necesita para su servicio. Al SuspendProcessessolicitar la suspensión del `Terminate` proceso, evitas que Auto Scaling intente cancelarlo por cualquier motivo, excepto por aquellos iniciados por una solicitud del usuario a la TerminateInstanceInAutoScalingGroupAPI.

Para obtener más información, consulte Suspensión y reanudación de procesos de Amazon EC2 Auto Scaling.

Limitaciones

> **Important**
>
> Al diseñar su aplicación en Amazon EC2 Auto Scaling para gestionar correctamente las terminaciones de instancias, tenga en cuenta las siguientes limitaciones.

Las instancias en mal estado eluden algunas protecciones

Si una instancia está en mal estado, Amazon EC2 Auto Scaling empezará a cancelarla incluso si cuenta con políticas de terminación personalizadas o protección escalable. La única forma de evitar que Auto Scaling sustituya las instancias en mal estado es suspender el `Terminate` proceso `HealthCheckReplaceUnhealthy`, o. Puede usar enlaces de ciclo de vida y una política de ciclo de vida de la instancia para permitir que la aplicación se cierre correctamente o copiar cualquier dato que necesite recuperar antes de que finalice la instancia en mal estado.

Característica	Controla las instancias en buen estado	Controla las instancias en mal estado
Políticas de rescisión personalizadas	Sí	No
Protección de reducción horizontal	Sí	No
Suspender `HealthCheck` o `Terminate` procesar `ReplaceUnhealthy`	Sí	Sí
Enlaces de ciclo de vida	Sí	Sí
Política del ciclo de vida de las	Sí	Sí

Los enlaces del ciclo de vida por sí solos no garantizan un cierre correcto

De forma predeterminada, los enlaces del ciclo de vida de las terminaciones funcionan en función del mejor esfuerzo posible. Si se suspende una acción del ciclo de vida de terminación, Amazon EC2 Auto Scaling procederá a terminar la instancia inmediatamente. Puede combinar los enlaces del ciclo de vida de la terminación con una política del ciclo de vida de la instancia para conservar las instancias cuando se abandonen las acciones relacionadas con el ciclo de vida de la terminación. Con esta combinación:

- Los enlaces del ciclo de vida de terminación intentan cerrar correctamente la aplicación después de que Auto Scaling active la finalización de una instancia y se complete el drenaje de cualquier balanceador de carga de Elastic Load Balancing configurado.
- Si se suspende una acción del ciclo de vida de terminación por cualquier motivo, la instancia pasa a un estado retenido en lugar de cerrarse.

- La instancia retenida permanece en su estado actual de Amazon EC2, lo que le permite completar manualmente los procedimientos de cierre o investigar el error.
- Puede finalizar manualmente las instancias retenidas llamando a la TerminateInstanceInAutoScalingGroupAPI después de completar las acciones necesarias.

Para obtener más información, consulte Controle la retención de instancias con políticas de ciclo de vida de las instancias.

En algunos casos, las opciones del mercado pueden interrumpirse con un aviso limitado

Si utiliza opciones del mercado de instancias, como instancias puntuales y reservas de capacidad interrumpible en su grupo de Auto Scaling, Amazon EC2 puede interrumpir y recuperar sus instancias en cualquier momento. Estas interrupciones eluden todos los mecanismos de protección de Amazon EC2 Auto Scaling, incluidos:

- Enlaces de ciclo de vida de terminación
- Protección contra la reducción horizontal de instancias
- Políticas de rescisión personalizadas
- Políticas de ciclo de vida de las
- Procesos suspendidos

Cuando una instancia puntual recibe una notificación de interrupción, dispone de aproximadamente dos minutos para realizar tareas de cierre sin problemas. Si bien puede utilizar los enlaces del ciclo de vida de terminación para responder a las interrupciones de las instancias puntuales, la instancia se cancelará forzosamente al final del período de dos minutos, incluso si el enlace del ciclo de vida aún está en curso. Las políticas del ciclo de vida de las instancias tampoco pueden evitar las interrupciones de las instancias puntuales.

Para obtener más información sobre la gestión de las interrupciones de instancias puntuales, consulte Interrupciones de instancias puntuales y prácticas recomendadas para Amazon EC2 Spot en la Guía del usuario de Amazon EC2.

Las terminaciones directas de Amazon EC2 eluden todas las protecciones

Si finaliza una instancia de su grupo de Auto Scaling directamente con la TerminateInstancesAPI de Amazon EC2, la terminación omite todos los mecanismos de protección de Auto Scaling de Amazon EC2.

Para terminar las instancias de su grupo de Auto Scaling y, al mismo tiempo, respetar las protecciones configuradas, utilice la TerminateInstanceInAutoScalingGroupAPI en su lugar.

Ejemplos de escenarios de

Cuando utiliza Amazon EC2 Auto Scaling, puede elegir la cantidad de gestión de flotas que Auto Scaling gestiona en su nombre en lugar del control directo que conserva sobre las decisiones de cancelación de instancias de EC2. Cuanto más sensible sea su carga de trabajo a las cancelaciones de instancias, más control querrá conservar. En los siguientes ejemplos, se describen las cargas de trabajo con diferentes niveles de tolerancia y las configuraciones recomendadas:

Ejemplo 1: Nodos de bases de datos distribuidas (tolerancia baja)

Ejecuta una base de datos distribuida en la que cada instancia de EC2 contiene una partición de los datos con un factor de replicación de 3. La pérdida de varias instancias que contienen réplicas de la misma partición podría provocar la pérdida de datos o hacer que la partición no esté disponible.

Desafío: Auto Scaling puede terminar las instancias más rápido de lo que se pueden volver a replicar los datos en otros nodos, y las terminaciones pueden reducir la capacidad por debajo de lo necesario para mantener el factor de replicación.

Tenga en cuenta las siguientes configuraciones:

- Habilite la protección escalable de instancias en todas las instancias de la base de datos; elimínela mediante programación solo después de confirmar que los datos se han replicado de forma segura en otros lugares.
- Configure los enlaces del ciclo de vida de las terminaciones con tiempos de espera prolongados en combinación con una política de ciclo de vida de las instancias para permitir que se complete la transferencia de datos.
- Establezca una política de mantenimiento de instancias con un porcentaje mínimo de mantenimiento del 100% para mantener la capacidad requerida.

Ejemplo 2: Procesamiento de trabajos de larga duración (tolerancia media)

Tiene una cola de Amazon SQS que recopila los mensajes entrantes para tareas de larga duración. Cuando llega un mensaje nuevo, una instancia de EC2 lo recupera e inicia un trabajo que tarda 3 horas en procesarse. A medida que la cola crece, Auto Scaling agrega instancias en función de sus políticas de escalado. A medida que la cola se reduce, Auto Scaling termina las instancias.

Desafío: Auto Scaling podría terminar una instancia que lleva 3 horas procesando un trabajo en lugar de una instancia inactiva. El trabajo se puede reiniciar en otra instancia, pero se pierde un progreso significativo.

Tenga en cuenta las siguientes configuraciones:

- Configure una política de terminación personalizada que priorice primero la finalización de las instancias inactivas.
- Utilice los ganchos del ciclo de vida de terminación para permitir que se completen los trabajos en curso.
- Activa la protección escalable de instancias mediante programación cuando una instancia inicie un trabajo y elimínala cuando se complete el trabajo.

Ejemplo 3: Flota de trabajadores para entornos de prueba (tolerancia alta)

Usted ejecuta una flota de instancias EC2 que ejecutan pruebas automatizadas, trabajos en CI/CD proceso o cargas de trabajo de desarrollo. Estas instancias de trabajo extraen tareas de una cola y los resultados de las pruebas se pueden regenerar si un trabajo falla.

Desafío: es posible que los trabajos de prueba se interrumpan durante los eventos de escalado, pero dado que las pruebas se pueden volver a intentar sin ningún impacto, es mejor optimizar los costos y la simplicidad en lugar de una disponibilidad sin interrupciones.

Tenga en cuenta las siguientes configuraciones:

- Utilice los ganchos del ciclo de vida de terminación para permitir que se completen las solicitudes durante el vuelo.
- Considere la posibilidad de utilizar instancias puntuales con una estrategia de asignación de capacidad optimizada para reducir aún más los costos.

Suspensión y reanudación de procesos de Amazon EC2 Auto Scaling

En este tema, se describe cómo suspender y reanudar uno o varios de los procesos del grupo de escalado automático para deshabilitar ciertas operaciones temporalmente.

Suspender los procesos puede resultar útil cuando se necesita investigar o solucionar un problema sin que las políticas de escalado o las acciones programadas interfieran. También ayuda a evitar

que Amazon EC2 Auto Scaling marque las instancias en mal estado y las reemplace mientras hace cambios en su grupo de escalado automático.

Temas

- Tipos de procesos
- Consideraciones para la suspensión de procesos
- Suspensión de procesos
- Reanudación de los procesos
- Efecto de la suspensión de procesos sobre otros procesos

> **Note**
>
> Además de las suspensiones que inicie, Amazon EC2 Auto Scaling también puede suspender los procesos de los grupos de Auto Scaling que repetidamente no consigan lanzar instancias. Esto es lo que se conoce como una suspensión administrativa. Una suspensión administrativa, en la mayoría de los casos, se aplica a los grupos de Auto Scaling que intentan lanzar instancias durante más de 24 horas, pero no lo logran. Puede reanudar los procesos suspendidos por Amazon EC2 Auto Scaling por razones administrativas.

Tipos de procesos

La función de reanudación-suspensión admite los siguientes procesos:

- `Launch`: agrega instancias al grupo de escalado automático cuando el grupo se escala horizontalmente o cuando Amazon EC2 Auto Scaling elige lanzar instancias por otros motivos, como cuando agrega instancias a un grupo de calentamiento.
- `Terminate`: elimina instancias del grupo de escalado automático cuando el grupo se reduce horizontalmente o cuando Amazon EC2 Auto Scaling elige finalizar instancias por otros motivos, como cuando una instancia finaliza por exceder su duración máxima de vida útil o no se realiza una comprobación de estado.
- `AddToLoadBalancer`: agrega instancias al grupo de destino del equilibrador de carga o equilibrador de carga clásico asociados cuando se lanzan. Para obtener más información, consulte Utilice Elastic Load Balancing para distribuir el tráfico entrante de la aplicación en su grupo de escalado automático..

- `AlarmNotification`— Acepta las notificaciones de CloudWatch las alarmas asociadas a las políticas de escalado dinámico. Para obtener más información, consulte Escalado dinámico para Amazon EC2 Auto Scaling.
- `AZRebalance`: equilibra el número de instancias de EC2 del grupo de manera uniforme en todas las zonas de disponibilidad especificadas cuando el grupo se desequilibra, por ejemplo, cuando una zona de disponibilidad que anteriormente no estaba disponible vuelve a un estado correcto. Para obtener más información, consulte Actividades de reequilibrio.
- `HealthCheck`: verifica el estado de las instancias y marca una instancia como en mal estado si Amazon EC2 o Elastic Load Balancing indica a Amazon EC2 Auto Scaling que la instancia está en mal estado. Este proceso puede invalidar el estado de una instancia que configura manualmente. Para obtener más información, consulte Comprobaciones de estado para instancias en un grupo de escalado automático.
- `InstanceRefresh`: termina y reemplaza instancias mediante la característica de actualización de instancias. Para obtener más información, consulte Uso de una actualización de instancias para actualizar instancias en un grupo de escalado automático.
- `ReplaceUnhealthy`: termina las instancias que están marcadas con un estado incorrecto y, a continuación, crea nuevas instancias para sustituirlas. Para obtener más información, consulte Comprobaciones de estado para instancias en un grupo de escalado automático.
- `ScheduledActions`: realiza las acciones de escalado programadas que ha creado o que se han creado automáticamente al crear un plan de escalado de AWS Auto Scaling y activa el escalado predictivo. Para obtener más información, consulte Escalado programado para Amazon EC2 Auto Scaling.

Consideraciones para la suspensión de procesos

Tenga en cuenta lo siguiente antes de suspender procesos:

- La suspensión `AlarmNotification` permite detener temporalmente las políticas de seguimiento, escalonamiento y escalado de objetivos del grupo sin eliminar las políticas de escalado ni las CloudWatch alarmas asociadas. Para detener temporalmente las políticas de escalado individuales, consulte Desactivación de una política de escalado para un grupo de escalado automático.
- Puede optar por suspender los procesos `HealthCheck` y `ReplaceUnhealthy` para reiniciar instancias sin que Amazon EC2 Auto Scaling termine las instancias en función de sus comprobaciones de estado. Sin embargo, si necesita Amazon EC2 Auto Scaling para continuar

realizando comprobaciones de estado de las instancias restantes, utilice la característica de espera. Para obtener más información, consulte Eliminación temporal de las instancias de un grupo de escalado automático.

- Si suspende los procesos `Launch`, `Terminate` o `AZRebalance`, y después realiza cambios en el grupo de escalado automático, por ejemplo, al desconectar instancias o cambiar las zonas de disponibilidad especificadas, el grupo puede desequilibrarse entre zonas de disponibilidad. Si esto sucede, después de reanudar los procesos suspendidos, Amazon EC2 Auto Scaling redistribuye gradualmente las instancias de manera uniforme entre las zonas de disponibilidad.
- Si suspendes el `Terminate` proceso, aún puedes forzar la cancelación de las instancias utilizando el delete-auto-scaling-groupcomando con la opción forzar la eliminación.
- La suspensión del proceso `Terminate` solo se aplica a las instancias que se encuentran actualmente en el estado `InService`. No impide la finalización de las instancias en otros estados, como `Pending`, o las instancias que no se reanuden correctamente luego de estar en espera.
- El `RemoveFromLoadBalancerLowPriority` proceso se puede ignorar cuando está presente en las llamadas para describir grupos de Auto Scaling utilizando la tecla AWS CLI o SDKs. Este proceso está en desuso y solo se conserva por motivos de compatibilidad con versiones anteriores.

Suspensión de procesos

Para suspender un proceso para un grupo de escalado automático, use uno de los siguientes métodos:

Console

Para suspender un proceso

1. Abra la consola Amazon EC2 en https://console.aws.amazon.com/ec2/y seleccione Auto Scaling Groups en el panel de navegación.
2. Seleccione la casilla situada junto al grupo de escalado automático.

   Se abre un panel dividido en la parte inferior de la página.
3. En la pestaña Details (Detalles) elija (Advanced configurations) Configuraciones avanzadas, Edit (Editar).
4. En Suspended processes (Procesos suspendidos), seleccione el proceso que desea suspender.

5. Elija Actualizar.

AWS CLI

Use el comando siguiente suspend-processes para suspender procesos individuales.

```
aws autoscaling suspend-processes --auto-scaling-group-name my-asg --scaling-
processes HealthCheck ReplaceUnhealthy
```

Para suspender todos los procesos, omita la opción `--scaling-processes` de la siguiente manera.

```
aws autoscaling suspend-processes --auto-scaling-group-name my-asg
```

Reanudación de los procesos

Para reanudar un proceso suspendido para un grupo de escalado automático, use uno de los siguientes métodos:

Console

Para reanudar un proceso suspendido

1. Abra la consola Amazon EC2 en https://console.aws.amazon.com/ec2/y seleccione Auto Scaling Groups en el panel de navegación.
2. Seleccione la casilla situada junto al grupo de escalado automático.

   Se abre un panel dividido en la parte inferior de la página.
3. En la pestaña Details (Detalles) elija (Advanced configurations) Configuraciones avanzadas, Edit (Editar).
4. Para Suspended processes (Procesos suspendidos), elimine el proceso suspendido.
5. Elija Actualizar.

AWS CLI

Para reanudar un proceso suspendido, utilice el siguiente comando resume-processes.

```
aws autoscaling resume-processes --auto-scaling-group-name my-asg --scaling-
processes HealthCheck
```

Para reanudar todos los procesos suspendidos, omita la opción `--scaling-processes` de la siguiente manera.

```
aws autoscaling resume-processes --auto-scaling-group-name my-asg
```

Efecto de la suspensión de procesos sobre otros procesos

Las siguientes secciones explican lo que ocurre cuando se suspenden los distintos tipos de procesos individualmente.

Temas

- Launch está suspendido
- Terminate está suspendido
- AddToLoadBalancer está suspendido
- AlarmNotification está suspendido
- AZRebalance está suspendido
- HealthCheck está suspendido
- InstanceRefresh está suspendido
- ReplaceUnhealthy está suspendido
- ScheduledActions está suspendido
- Consideraciones adicionales

Launch está suspendido

- `AlarmNotification` sigue activo, pero el grupo de escalado automático no puede iniciar actividades de escalado horizontal para alarmas infractoras.
- `ScheduledActions` está activo, pero el grupo de escalado automático no puede iniciar actividades de escalado horizontal para ninguna acción programada que se produzca.
- `AZRebalance` deja de reequilibrar el grupo.

- `ReplaceUnhealthy` continúa finalizando instancias en mal estado, pero no lanza reemplazos. Cuando reanude el proceso `Launch`, Amazon EC2 Auto Scaling reemplaza inmediatamente las instancias que finalizó durante el tiempo en que se suspendió `Launch`.
- `InstanceRefresh` no reemplaza las instancias.

`Terminate` está suspendido

- `AlarmNotification` sigue activo, pero el grupo de escalado automático no puede iniciar actividades de reducción horizontal para alarmas infractoras.
- `ScheduledActions` está activo, pero el grupo de escalado automático no puede iniciar actividades de reducción horizontal para ninguna acción programada que se produzca.
- `AZRebalance` sigue activo, pero no funciona correctamente. Puede lanzar nuevas instancias sin terminar las antiguas. Esto puede provocar que su grupo de escalado automático aumente hasta un 10 % más que su tamaño máximo, ya que se permite que ocurra esto durante las actividades de reequilibrado. Su grupo de escalado automático podría permanecer por encima de su tamaño máximo hasta que se reanude el proceso `Terminate`.
- `ReplaceUnhealthy` está inactivo, pero no `HealthCheck`. Cuando se reanuda `Terminate`, el proceso `ReplaceUnhealthy` empieza a ejecutarse inmediatamente. Si las instancias se marcaron como en mal estado mientras `Terminate` se encuentra suspendido, se reemplazarán de inmediato.
- `InstanceRefresh` no reemplaza las instancias.

`AddToLoadBalancer` está suspendido

- Amazon EC2 Auto Scaling lanza las instancias, pero no las agrega al grupo de destino del balanceador de carga o al Classic Load Balancer. Cuando se reanuda el proceso `AddToLoadBalancer`, se reanuda la adición de instancias al balanceador de carga cuando se lanzan. Sin embargo, no se añaden las instancias que se lanzaron mientras este proceso estaba suspendido. Debe registrar dichas instancias manualmente.

`AlarmNotification` está suspendido

- Amazon EC2 Auto Scaling no invoca políticas de escalado cuando se infringe CloudWatch un umbral de alarma. Al reanudar `AlarmNotification`, Amazon EC2 Auto Scaling tiene en cuenta las políticas con umbrales de alarma que se han interrumpido.

AZRebalance está suspendido

- Amazon EC2 Auto Scaling no intenta redistribuir instancias tras determinados eventos. Sin embargo, si se produce un evento de escalado o reducción horizontales, el proceso de escalado intenta equilibrar igualmente las zonas de disponibilidad. Por ejemplo, durante el escalado horizontal, se lanzan las instancias en la zona de disponibilidad con el menor número de instancias. Si el grupo se desequilibra durante la suspensión de `AZRebalance` y lo reanuda, Amazon EC2 Auto Scaling intenta reequilibrar el grupo. En primer lugar, llama a `Launch` y, a continuación, a `Terminate`.
- Los grupos preparados no se ven afectados cuando `AZRebalance` está suspendida.

HealthCheck está suspendido

- Amazon EC2 Auto Scaling deja de marcar las instancias con un estado incorrecto como resultado de las comprobaciones de estado de EC2 y Elastic Load Balancing. Las comprobaciones de estado personalizadas siguen funcionando correctamente. Tras suspender `HealthCheck`, si es necesario, puede configurar manualmente el estado de las instancias del grupo y que `ReplaceUnhealthy` las sustituya.

InstanceRefresh está suspendido

- Amazon EC2 Auto Scaling deja de reemplazar instancias debido a una actualización de instancias. Si hay una actualización de instancias en curso, se detiene la operación sin cancelarla.

ReplaceUnhealthy está suspendido

- Amazon EC2 Auto Scaling deja de reemplazar las instancias marcadas con un estado incorrecto. Las instancias que no superan las comprobaciones de estado de EC2 o Elastic Load Balancing seguirán estando marcadas con un estado incorrecto. En cuanto se reanuda el proceso `ReplaceUnhealthy`, Amazon EC2 Auto Scaling reemplaza las instancias marcadas con un estado incorrecto durante la suspensión de este proceso. El proceso `ReplaceUnhealthy` llama primero a `Terminate` y después `Launch`.

`ScheduledActions` está suspendido

- Amazon EC2 Auto Scaling no pone en marcha acciones programadas que están programadas para su activación durante el periodo de suspensión. Cuando se reanuda `ScheduledActions`, Amazon EC2 Auto Scaling solo considera acciones programadas cuyo tiempo programado aún no ha pasado.

Consideraciones adicionales

Además, cuando `Launch` o `Terminate` están suspendidos, es posible que las siguientes funciones no trabajen correctamente:

- Duración máxima de la instancia: cuando `Launch` o `Terminate` se suspenden, la característica de duración máxima de instancia no puede reemplazar ninguna instancia.
- Interrupciones de instancias de spot: si `Terminate` se suspende y el grupo de escalado automático tiene instancias de spot, todavía pueden terminar en caso de que la capacidad de spot ya no esté disponible. Durante la suspensión de `Launch`, Amazon EC2 Auto Scaling no puede lanzar instancias de reemplazo de otro grupo de instancias de spot o del mismo grupo de instancias de spot cuando vuelve a estar disponible.
- Reequilibrio de la capacidad: si se suspende `Terminate` y utiliza el reequilibrio de la capacidad para gestionar las interrupciones de instancias de spot, el servicio de spot de Amazon EC2 aún puede terminar instancias en caso de que la capacidad del spot ya no esté disponible. Si se suspende `Launch`, Amazon EC2 Auto Scaling no puede lanzar instancias de reemplazo de otro grupo de instancias de spot o del mismo grupo de instancias de spot cuando vuelve a estar disponible.
- Asociación y desconexión de instancias: cuando `Launch` y `Terminate` se suspenden, puede desconectar las instancias asociadas al grupo de escalado automático, pero mientras `Launch` está suspendido, no puede asociar nuevas instancias al grupo.
- Instancias en espera: cuando `Launch` y `Terminate` están suspendidos, puede poner una instancia en el estado `Standby`, pero mientras `Launch` está suspendido, no se puede reactivar una instancia con el estado `Standby`.

Supervisión de sus grupos de Amazon EC2 Auto Scaling

La supervisión es una parte importante del mantenimiento de la confiabilidad, la disponibilidad y el rendimiento de Amazon EC2 Auto Scaling y sus Nube de AWS soluciones. AWS proporciona las siguientes herramientas de supervisión para observar Amazon EC2 Auto Scaling, informar cuando algo va mal y tomar medidas automáticas cuando sea necesario:

Comprobaciones de estado

Amazon EC2 Auto Scaling realiza periódicamente comprobaciones de estado de las instancias de su grupo de Auto Scaling. Si una instancia no supera su comprobación de estado, se marca como en mal estado y se terminará mientras Amazon EC2 Auto Scaling lanza una nueva instancia para reemplazarla. Para obtener más información, consulte Comprobaciones de estado para instancias en un grupo de escalado automático.

Panel de AWS Health

Panel de estado Muestra información y también proporciona notificaciones que se invocan cuando se producen cambios en el estado de AWS los recursos. La información se presenta de dos formas: en un panel donde se muestran los eventos recientes y próximos organizados por categorías, y en un registro de eventos que contiene todos los eventos de los últimos 90 días. Para obtener más información, consulte Panel de estado notificaciones para Amazon EC2 Auto Scaling.

CloudTrail

Con AWS CloudTrail, puede realizar un seguimiento de las llamadas realizadas a la API Auto Scaling de Amazon EC2 por su parte o en su nombre. Cuenta de AWS CloudTrail almacena la información en archivos de registro en el bucket de Amazon S3 que especifique. Puede utilizar estos archivos de registro para monitorear la actividad de los grupos de Auto Scaling. Los registros incluyen las solicitudes que se han realizado, las direcciones IP de origen de las que proceden las solicitudes, quién ha efectuado la solicitud, cuándo se ha realizado, etc. Para obtener más información, consulte Registre las llamadas a la API Amazon EC2 Auto Scaling con AWS CloudTrail.

> **Recopilación de registros de sus instancias de Amazon EC2**
>
> Puede utilizarlos CloudWatch para recopilar registros de los sistemas operativos para sus instancias EC2. Para obtener más información, consulte Recopilar métricas y registros de

instancias de Amazon EC2 y servidores locales con el CloudWatch agente y Ver los datos de registro enviados a CloudWatch Logs en la Guía del usuario de Amazon CloudWatch . Para obtener información sobre otros AWS servicios que pueden ayudarlo a registrar y recopilar datos sobre sus cargas de trabajo, consulte la guía de registro y monitoreo para propietarios de aplicaciones en la AWS Guía prescriptiva.

Amazon CloudWatch

Amazon le CloudWatch ayuda a analizar los registros y, en tiempo real, a supervisar las métricas de sus AWS recursos y aplicaciones alojadas. Puede recopilar métricas y realizar un seguimiento de las métricas, crear paneles personalizados y definir alarmas que le advierten o que toman medidas cuando una métrica determinada alcanza el umbral que se especifique. Por ejemplo, puede recibir una notificación cuando la actividad de la red de repente sea superior o inferior al valor esperado de una métrica. Para obtener más información sobre el uso de este servicio para supervisar las métricas de sus instancias y grupos de escalado automático, consulte Supervise CloudWatch las métricas de sus grupos e instancias de Auto Scaling.

CloudWatch también realiza un seguimiento AWS de las métricas de uso de API para Amazon EC2 Auto Scaling. Puede utilizar estas métricas para configurar alarmas que avisen cuando el volumen de llamadas a la API infrinja un límite definido. Para obtener más información, consulta las estadísticas AWS de uso en la Guía del CloudWatch usuario de Amazon.

AWS Compute Optimizer

Compute Optimizer proporciona recomendaciones de instancias de Amazon EC2 que le pueden ayudar a decidir si pasar a un tipo de instancia nuevo. Analiza si el tipo de instancia de un grupo de escalado automático es óptimo y genera recomendaciones para reducir el costo y mejorar el rendimiento de sus cargas de trabajo. Para obtener más información, consulte Obtenga recomendaciones de tipos de instancias con AWS Compute Optimizer.

Amazon EventBridge

Amazon EventBridge es un servicio de bus de eventos sin servidor que facilita la conexión de sus aplicaciones con datos de diversas fuentes. EventBridge ofrece un flujo de datos en tiempo real desde sus propias aplicaciones, aplicaciones Software-as-a-Service (SaaS) y AWS servicios, y dirige esos datos a destinos como Lambda. Esto le permite monitorear los eventos que ocurren en los servicios y crear arquitecturas basadas en eventos. Para obtener más información, consulte Se usa EventBridge para gestionar eventos de Auto Scaling.

AWS Security Hub CSPM

Utilice AWS Security Hub CSPM para monitorear el uso de Amazon EC2 Auto Scaling en relación con las mejores prácticas de seguridad. Security Hub CSPM utiliza controles de seguridad de detección para evaluar las configuraciones de los recursos y los estándares de seguridad para ayudarlo a cumplir con varios marcos de cumplimiento. Para obtener más información sobre el uso de Security Hub CSPM para evaluar los recursos de Auto Scaling de Amazon EC2, consulte los controles de Auto Scaling de Amazon EC2 en la Guía del usuario.AWS Security Hub

Amazon Simple Notification Service

Puede configurar los grupos de Auto Scaling para enviar notificaciones de Amazon SNS cuando Amazon EC2 Auto Scaling lanza o termina instancias. Para obtener más información, consulte Opciones de notificaciones de Amazon SNS para Amazon EC2 Auto Scaling.

Comprobaciones de estado para instancias en un grupo de escalado automático

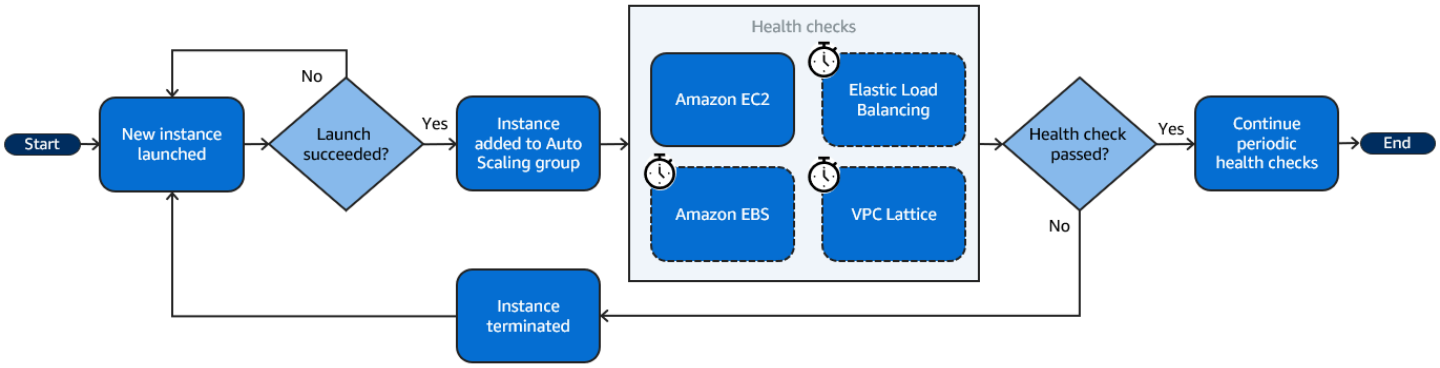
Amazon EC2 Auto Scaling supervisa de manera continua el estado de instancias de un grupo de escalado automático para mantener la capacidad deseada.

Todas las instancias de un grupo de escalado automático comienzan con un estado `Healthy`. Se entiende que las instancias están en buen estado, a menos que Amazon EC2 Auto Scaling reciba una notificación de que están en mal estado. Puede recibir notificaciones de diversas fuentes cuando una instancia deja de estar en buen estado y debe reemplazarse. Estos orígenes pueden ser:

- Amazon EC2
- Elastic Load Balancing
- VPC Lattice
- Amazon EBS
- Comprobaciones de estado personalizadas que defina

Cuando Amazon EC2 Auto Scaling determina que una instancia `InService` no está en buen estado, la reemplaza por una nueva instancia para mantener la capacidad deseada del grupo. La nueva instancia se lanza con la configuración actual del grupo de escalado automático y su plantilla de lanzamiento o configuración de lanzamiento asociada.

En el siguiente diagrama de flujo se ilustra el proceso de inicialización de una nueva instancia en un grupo de escalado automático. Comienza con la inicialización de la instancia. Si la inicialización se realiza correctamente, la instancia se agrega al grupo de escalado automático. A continuación, Amazon EC2 Auto Scaling realiza comprobaciones de estado en la instancia mediante las comprobaciones de estado integradas de Amazon EC2 y, después de un período de gracia, cualquier comprobación de estado opcional que haya habilitado para el grupo. Estas comprobaciones de estado se realizan periódicamente. Si alguna de las comprobaciones de estado falla, se reemplaza la instancia.



Las instancias en mal estado también se pueden producir cuando una instancia termina inesperadamente, por ejemplo, debido a una interrupción de una instancia de spot o a una terminación manual por parte de un usuario. De nuevo, Amazon EC2 Auto Scaling inicializará automáticamente una instancia de reemplazo en estos casos para mantener la capacidad deseada.

Contenido

- Acerca de las comprobaciones de estado para el grupo de escalado automático
- Establezca el periodo de gracia de la comprobación de estado para un grupo de escalado automático
- Supervisión de las instancias de escalado automático con volúmenes de Amazon EBS dañados mediante comprobaciones de estado
- Configuración de una comprobación de estado personalizada para el grupo de escalado automático
- Visionado del motivo de los errores de una comprobación de estado
- Utilice las comprobaciones de estado de las aplicaciones con un grupo de Auto Scaling
- Solución de problemas de instancias en mal estado en Amazon EC2 Auto Scaling

Acerca de las comprobaciones de estado para el grupo de escalado automático

En este tema se ofrece información general de los tipos de comprobaciones de estado disponibles y se describen las consideraciones clave para integrar las comprobaciones de estado de Amazon EC2 Auto Scaling con sus aplicaciones.

Contenido

- Tipos de comprobación de estado
- Comprobaciones de estado de Amazon EC2
- Controles de estado de Elastic Load Balancing
- Comprobaciones de estado de VPC Lattice
- Cómo Amazon EC2 Auto Scaling minimiza el tiempo de inactividad
- Comprobaciones de estado para las instancias en un grupo de calentamiento
- Consideraciones acerca de las comprobaciones de estado

Tipos de comprobación de estado

Amazon EC2 Auto Scaling puede determinar el estado de una instancia `InService` mediante una o más de las siguientes comprobaciones de estado:

Tipo de comprobación de estado	¿Qué comprueba?
Comprobaciones de estado y eventos programados de Amazon EC2	• Comprueba que la instancia se esté ejecutando. • Comprueba si hay problemas subyacentes de hardware o software que podrían perjudicar la instancia. Este es el tipo de comprobación de estado predeterminado para un grupo de escalado automático.
Controles de estado de Elastic Load Balancing	• Comprueba si el equilibrador de carga informa que la instancia está en buen estado y confirma si la instancia está disponible para gestionar solicitudes.

Tipo de comprobación de estado	¿Qué comprueba?
	Para ejecutar este tipo de comprobación de estado, debe activarla para el grupo de escalado automático.
Comprobaciones de estado de VPC Lattice	• Comprueba si VPC Lattice informa que la instancia está en buen estado y confirma si la instancia está disponible para gestionar solicitudes. Para ejecutar este tipo de comprobación de estado, debe activarla para el grupo de escalado automático.
Comprobaciones de estado de Amazon EBS	• Comprueba si se puede acceder a los volúmenes de EBS y pasa las comprobaciones de estado. I/O Para ejecutar este tipo de comprobación de estado, debe activarla para el grupo de escalado automático.
Comprobaciones de estado personalizadas	• Comprueba si hay otros problemas que puedan indicar incidencias en el estado de las instancias, según las comprobaciones de estado personalizadas.

Comprobaciones de estado de Amazon EC2

Cuando se lanza una instancia, se adjunta al grupo de escalado automático e ingresa en el estado `InService`. Para obtener información sobre los distintos estados del ciclo de vida de las instancias de un grupo de escalado automático, consulte Ciclo de vida de instancias de Amazon EC2 Auto Scaling.

Amazon EC2 Auto Scaling comprueba periódicamente el estado de todas las instancias dentro del grupo de escalado automático para asegurarse de que se ejecuten y estén en buenas condiciones.

Las comprobaciones de estado

Amazon EC2 Auto Scaling utiliza los resultados de las comprobaciones de estado de instancias de Amazon EC2 y las comprobaciones de estado del sistema para determinar el estado de una instancia. Si la instancia se encuentra en cualquier estado de Amazon EC2 distinto de `running`

o si el estado de las comprobaciones de estado se vuelve `impaired`, Amazon EC2 Auto Scaling considera que la instancia está en mal estado y la reemplaza. Esto ocurre cuando la instancia tiene alguno de los estados siguientes:

- `stopping`
- `stopped`
- `shutting-down`
- `terminated`

Las comprobaciones de estado de Amazon EC2 no requieren ninguna configuración especial y siempre están habilitadas. Para obtener más información, consulte Tipos de comprobaciones de estado en la Guía del usuario de Amazon EC2.

> **Important**
>
> Amazon EC2 Auto Scaling permite que las comprobaciones de estado fallen ocasionalmente, sin realizar ninguna acción. Cuando se produce un error en una comprobación de estado, Amazon EC2 Auto Scaling espera unos minutos AWS para solucionar el problema. No marca de inmediato una instancia como `Unhealthy` cuando el estado de las comprobaciones de estado se vuelve `impaired`. Además, EC2 Auto Scaling no marca la instancia como `Unhealthy` si una comprobación de estado devuelve `insufficient-data`.
> Sin embargo, si Amazon EC2 Auto Scaling detecta que una instancia ya no se encuentra en el estado `running`, esta situación se trata como un error inmediato. En este caso, marca de inmediato la instancia como `Unhealthy` y la reemplaza.

Verificaciones del estado de la aplicación

Las comprobaciones del estado de las aplicaciones extienden la señal de verificación del estado de Amazon EC2 a su aplicación. Supervisan si una aplicación de su instancia responde a las llamadas HTTP o HTTPS. Cuando una instancia de su grupo de Auto Scaling tiene comprobaciones de estado de la aplicación asociadas e incluidas en la agregación, Amazon EC2 Auto Scaling evalúa el estado general de la aplicación. Si el estado se notifica`impaired`, Amazon EC2 Auto Scaling considera que la instancia no está en buen estado y la reemplaza. No se requiere ninguna configuración de grupo adicional de Auto Scaling.

Las comprobaciones del estado de las aplicaciones se crean y administran en Amazon EC2, no en Amazon EC2 Auto Scaling. Para obtener más información sobre la configuración de las comprobaciones del estado de las solicitudes para su grupo, consulte. Utilice las comprobaciones de estado de las aplicaciones con un grupo de Auto Scaling

Eventos programados

Amazon EC2 puede programar ocasionalmente los eventos de las instancias para que se ejecuten después de una marca temporal determinada. Para obtener más información, consulte Eventos programados para las instancias en la Guía del usuario de Amazon EC2.

Si una de las instancias se ve afectada por un evento programado, Amazon EC2 Auto Scaling considera que la instancia está en mal estado y la reemplaza. La instancia no empieza a cerrarse hasta que se alcanza la fecha y hora especificadas en la marca temporal.

Controles de estado de Elastic Load Balancing

Cuando habilita las comprobaciones de estado de Elastic Load Balancing para su grupo de escalado automático, Amazon EC2 Auto Scaling puede usar los resultados de esas comprobaciones de estado para determinar el estado de una instancia.

Para poder habilitar las comprobaciones de estado de Elastic Load Balancing para el grupo de escalado automático, debe configurar un equilibrador de carga de Elastic Load Balancing y una comprobación de estado para determinar si las instancias están en buen estado. Para obtener más información, consulte Preparación para asociar un equilibrador de carga de Elastic Load Balancing.

Después de asociar el equilibrador de carga al grupo de escalado automático, ocurre lo siguiente:

- Amazon EC2 Auto Scaling registra las instancias del grupo de escalado automático con el equilibrador de carga.
- Una vez que se termina de registrar una instancia, ingresa al estado `InService` y estará disponible para su uso con el equilibrador de carga.

De forma predeterminada, Amazon EC2 Auto Scaling ignora los resultados de las comprobaciones de estado de Elastic Load Balancing. Después de activar estas comprobaciones de estado para su grupo de escalado automático, cuando Elastic Load Balancing informa de una instancia registrada como `Unhealthy`, Amazon EC2 Auto Scaling marca la instancia como `Unhealthy` en su próxima comprobación de estado periódica y la reemplaza.

Si el drenaje de conexiones (retardo de anulación del registro) está habilitado para su equilibrador de carga, Amazon EC2 Auto Scaling espera a que se completen las solicitudes en curso o que expire el tiempo de espera máximo antes de terminar las instancias en mal estado.

> **Note**
>
> Para obtener instrucciones sobre cómo asociar el equilibrador de carga y activar las comprobaciones de estado de Elastic Load Balancing para el grupo de escalado automático, consulte Asociación de un equilibrador de carga de Elastic Load Balancing al grupo de escalado automático.
> Cuando activa las comprobaciones de estado de Elastic Load Balancing para un grupo, Amazon EC2 Auto Scaling puede reemplazar las instancias notificadas por Elastic Load Balancing como en mal estado, pero solo después de que el equilibrador de carga se encuentre en estado `InService`. Para obtener más información, consulte Verificación del estado de asociación del equilibrador de carga.

Comprobaciones de estado de VPC Lattice

De forma predeterminada, Amazon EC2 Auto Scaling ignora los resultados de las comprobaciones de estado de VPC Lattice. Como opción, puede activar las comprobaciones de estado para el grupo de escalado automático. Después de hacer esto, cuando VPC Lattice informa de una instancia registrada como `Unhealthy`, Amazon EC2 Auto Scaling marca la instancia como `Unhealthy` en su próxima comprobación de estado periódica y la reemplaza. El proceso de registro de las instancias y, a continuación, comprobar su estado es el mismo que el de las comprobaciones de estado de Elastic Load Balancing.

> **Note**
>
> Para obtener instrucciones sobre cómo asociar el grupo de destino de VPC Lattice y activar las comprobaciones de estado de VPC Lattice para el grupo de escalado automático, consulte Asociación de un grupo de destino de VPC Lattice a su grupo de escalado automático.
> Cuando activa las comprobaciones de estado de VPC Lattice para un grupo, Amazon EC2 Auto Scaling puede reemplazar las instancias notificadas por VPC Lattice como en mal estado, pero solo después de que el grupo de destino se encuentre en estado `InService`.

Para obtener más información, consulte Verificación del estado de asociación de su grupo de destino de VPC Lattice.

Cómo Amazon EC2 Auto Scaling minimiza el tiempo de inactividad

De modo predeterminado, las instancias nuevas se aprovisionan al mismo tiempo que se terminan las instancias existentes, lo que puede impedir que se acepten nuevas solicitudes hasta que las instancias nuevas estén totalmente operativas.

Si Amazon EC2 Auto Scaling determina que alguna instancia ya no se pone en marcha (o se marcó como `Unhealthy` con el comando set-instance-health), la reemplaza de inmediato. Sin embargo, si se detecta que otras instancias no están en buen estado, Amazon EC2 Auto Scaling utiliza el siguiente enfoque para recuperarse de errores. Este enfoque minimiza cualquier tiempo de inactividad que pueda producirse debido a problemas temporales o comprobaciones de estado mal configuradas.

- Si una actividad de escalado está en curso y el grupo de escalado automático es inferior a la capacidad deseada en un 10 % o más, Amazon EC2 Auto Scaling espera la actividad de escalado en curso antes de reemplazar las instancias en mal estado.
- Cuando se escala horizontalmente, Amazon EC2 Auto Scaling espera a que las instancias pasen una comprobación de estado inicial. También espera a que finalice el calentamiento de instancias predeterminado para asegurarse de que las nuevas instancias estén listas.
- Cuando las instancias terminen de calentarse y el grupo haya aumentado a más del 90 % de la capacidad deseada, Amazon EC2 Auto Scaling reemplaza las instancias en mal estado de la siguiente manera:
  - Amazon EC2 Auto Scaling solo reemplaza hasta el 10 % de la capacidad deseada del grupo a la vez. Lo hace hasta que se sustituyan todas las instancias en mal estado.
  - Al reemplazar instancias, espera a que las nuevas instancias pasen una comprobación de estado inicial. También espera a que finalice el calentamiento de instancias predeterminado antes de continuar.

Note

- Si el tamaño de un grupo de escalado automático es lo suficientemente pequeño como para que el valor resultante del 10 % sea inferior a uno, Amazon EC2 Auto Scaling

reemplaza las instancias en mal estado de una en una. Esto podría provocar tiempo de inactividad para el grupo.

- Puede modificar el valor predeterminado del 10 por ciento al establecer una política de mantenimiento de instancias para cambiar la velocidad a la que Auto Scaling reemplaza las instancias en mal estado. Sin embargo, es posible que Auto Scaling siga reduciendo la velocidad a la que marca las instancias como en mal estado.

  Por ejemplo, si las comprobaciones de estado de Elastic Load Balancing informan de que todas las instancias de un grupo de escalado automático están en mal estado y el equilibrador de carga se encuentra en el estado `InService`, Amazon EC2 Auto Scaling puede marcar menos instancias en mal estado a la vez. Esto puede dar lugar a que se reemplacen muchas menos instancias a la vez que el 10 % aplicado en otros escenarios. Esto le proporciona tiempo para corregir el problema sin que Amazon EC2 Auto Scaling termine de manera automática todo el grupo.

Comprobaciones de estado para las instancias en un grupo de calentamiento

Amazon EC2 Auto Scaling también realiza comprobaciones de estado de instancias en un grupo de calentamiento. Para obtener más información, consulte Visualización del estado de la comprobación de estado y el motivo de los errores de la comprobación de estado.

Consideraciones acerca de las comprobaciones de estado

A continuación se indican las consideraciones a tener en cuenta al utilizar las comprobaciones de estado de Amazon EC2 Auto Scaling.

- Si necesita que suceda algo en la instancia que está terminando o en la instancia que se está iniciando, puede usar enlaces de ciclo de vida. Estos enlaces permiten realizar una acción personalizada a medida que Amazon EC2 Auto Scaling lanza o termina instancias. Para obtener más información, consulte Enlaces de ciclo de vida de Amazon EC2 Auto Scaling.
- Amazon EC2 Auto Scaling no ofrece una forma de eliminar las comprobaciones de estado de Amazon EC2 y los eventos programados de sus comprobaciones de estado. Si no desea que se reemplacen las instancias, le recomendamos suspender el proceso `ReplaceUnhealthy` y `HealthCheck` para los grupos de Auto Scaling individuales. Para obtener más información, consulte Suspensión y reanudación de procesos de Amazon EC2 Auto Scaling.
- Puede intentar utilizar el comando set-instance-health para establecer manualmente el estado de la instancia en un mal estado de vuelta como `Healthy`. Si recibe un error, probablemente

se deba a que la instancia ya está terminando. En general, volver a establecer el estado de una instancia en `Healthy` con el comando set-instance-health solo es útil en los casos en que el proceso `ReplaceUnhealthy` o el proceso `Terminate` está suspendido.

- Si necesita solucionar un problema de una instancia sin que las comprobaciones de estado interfieran, puede poner la instancia en estado `Standby`. Amazon EC2 Auto Scaling no realiza comprobaciones de estado en las instancias que se encuentran en estado `Standby`hasta que vuelva a poner la instancia en servicio. Para obtener más información, consulte Eliminación temporal de las instancias de un grupo de escalado automático.
- Cuando se termina la instancia, todas las direcciones IP elásticas asociadas se desasocian y no se asocian automáticamente a la nueva instancia. Tiene que asociar las direcciones IP elásticas a la instancia nueva, o hacerlo automáticamente mediante una solución basada en un enlace de ciclo de vida. Para obtener más información, consulte Direcciones IP elásticas en la Guía del usuario de Amazon EC2.
- Del mismo modo, cuando se termina la instancia, los volúmenes de EBS adjuntos se desconectan (o eliminan, según el atributo `DeleteOnTermination` del volumen). Tiene que adjuntar estos volúmenes de EBS a la nueva instancia, o hacerlo automáticamente con una solución basada en un enlace de ciclo de vida. Para obtener más información, consulte Adjunte un volumen de Amazon EBS a una instancia en la Guía del usuario de Amazon EBS.

Establezca el periodo de gracia de la comprobación de estado para un grupo de escalado automático

Cuando una comprobación de estado de Amazon EC2 Auto Scaling determina que una instancia `InService` no está en buen estado, la reemplaza por una nueva instancia. El periodo de gracia de la comprobación de estado especifica la cantidad mínima de tiempo (en segundos) para mantener una nueva instancia en servicio antes de que la finalice si no está en buen estado.

Amazon EC2 Auto Scaling podría necesitar un caso de uso de ejemplo para evitar tomar medidas si las comprobaciones de estado de Elastic Load Balancing tienen errores y la causa es que la instancia aún se está inicializando. Las comprobaciones de estado de Elastic Load Balancing se ejecutan en paralelo y se inician cuando la instancia se registra en el equilibrador de carga. El período de gracia evita que Amazon EC2 Auto Scaling marque las instancias recién inicializadas como `Unhealthy` y las cancele de manera innecesaria si no pasan estas comprobaciones de estado inmediatamente después de entrar en el estado `InService`.

De forma predeterminada, el período de gracia de la comprobación de estado es de 300 segundos cuando se crea un grupo de escalado automático. Su valor predeterminado es de 0 segundos cuando se crea un grupo de Auto Scaling mediante el SDK AWS CLI o un SDK. Un valor de 0 desactiva el periodo de gracia de la comprobación de estado.

Si este valor es demasiado alto, se reduce la eficacia de las comprobaciones de estado de Amazon EC2 Auto Scaling. Si utiliza un enlace de ciclo de vida para el lanzamiento de instancias, puede configurar el período de gracias de la comprobación de estado en 0. Gracias a los enlaces de ciclo de vida, Amazon EC2 Auto Scaling ofrece una forma de garantizar que las instancias se inicialicen siempre antes de que entren en el estado `InService`. Para obtener más información, consulte Enlaces de ciclo de vida de Amazon EC2 Auto Scaling.

El período de gracia se aplica a las siguientes instancias:

- Instancias recién lanzadas
- Instancias que se vuelven a poner en servicio después de estar en modo de espera
- Instancias que se adjuntan manualmente al grupo

> **Important**
>
> Durante el periodo de gracia de la comprobación de estado, si Amazon EC2 Auto Scaling detecta que una instancia ya no se encuentra en el estado `running` de Amazon EC2, la marca como `Unhealthy` y la reemplaza. Por ejemplo, si detiene una instancia de un grupo de escalado automático, se marcará como `Unhealthy` y se reemplazará.

Establezca el período de gracia de la comprobación de estado para un grupo

Puede establecer el período de gracia de la comprobación de estado para grupos de escalado automático.

Console

Cómo modificar el período de gracia de la comprobación de estado de un grupo nuevo

Cuando cree el grupo de escalado automático, ingrese la cantidad de tiempo (en segundos) en la página Configure las opciones avanzadas, Comprobaciones de estado, Período de gracia de la comprobación de estado. Este es el tiempo que Amazon EC2 Auto Scaling debe esperar antes de comprobar el estado de una instancia una vez que pasa al estado `InService`.

AWS CLI

Cómo modificar el período de gracia de la comprobación de estado de un grupo nuevo

Agregue la opción `--health-check-grace-period` al comando create-auto-scaling-group. En el siguiente ejemplo se configura el período de gracia de la comprobación de estado con un valor de *60* segundos para un nuevo grupo de escalado automático denominado *my-asg*.

```
aws autoscaling create-auto-scaling-group --auto-scaling-group-name my-asg \
  --health-check-grace-period 60 ...
```

Console

Cómo modificar el período de gracia de la comprobación de estado de un grupo existente

1. Abra la consola de Amazon EC2 en https://console.aws.amazon.com/ec2/ y seleccione Auto Scaling Groups en el panel de navegación.
2. En la barra de navegación de la parte superior de la pantalla, elija la misma Región de AWS que utilizó cuando creó el grupo de escalado automático.
3. Seleccione la casilla situada junto al grupo de escalado automático.

   Se abre un panel dividido en la parte inferior de la página.
4. En la pestaña Details (Detalles), elija Health checks (Comprobaciones de estado), Edit (Editar).
5. En Health check grace period (Período de gracia de comprobación de estado), ingrese el tiempo, en segundos. Este es el tiempo que Amazon EC2 Auto Scaling debe esperar antes de comprobar el estado de una instancia una vez que pasa al estado `InService`.
6. Elija Actualizar.

AWS CLI

Cómo modificar el período de gracia de la comprobación de estado de un grupo existente

Agregue la opción `--health-check-grace-period` al comando update-auto-scaling-group. En el siguiente ejemplo se configura el período de gracia de la comprobación de estado con un valor de *120* segundos para un grupo de escalado automático existente denominado *my-asg*.

```
aws autoscaling update-auto-scaling-group --auto-scaling-group-name my-asg \
```

```
--health-check-grace-period 120
```

> **Note**
>
> También le recomendamos encarecidamente establecer el tiempo de preparación predeterminado de instancias para el grupo de escalado automático. Para obtener más información, consulte Establecimiento de la preparación predeterminada de instancias para un grupo de escalado automático.

Supervisión de las instancias de escalado automático con volúmenes de Amazon EBS dañados mediante comprobaciones de estado

Puede activar las comprobaciones de estado de Amazon EBS para el grupo de escalado automático para asegurarse de que Amazon EC2 Auto Scaling supervisa todo el sistema en el que se ejecuta la aplicación.

Después de activar las comprobaciones de estado, Amazon EC2 Auto Scaling recibe los resultados de las comprobaciones de estado de Amazon EC2 realizadas en los volúmenes de EBS asociados a una instancia. Si no se puede acceder a un volumen o no pasa las comprobaciones de I/O estado, la comprobación de estado fallará y la instancia correspondiente se considerará en mal estado. Cuando Amazon EC2 Auto Scaling detecta que una instancia no está en buen estado, la reemplaza.

En este tema se presupone que está familiarizado con las comprobaciones de estado de EBS adjuntas. Si no lo está, consulte la sección Comprobaciones de estado de EBS adjuntas de la Guía del usuario de Amazon EC2 para obtener más información. En el siguiente tema se describe cómo activar las comprobaciones de estado de Amazon EC2 Auto Scaling que se basan en las comprobaciones de estado de EBS adjuntas.

> **Note**
>
> Puede activar las comprobaciones de estado de Amazon EBS para todos los grupos de escalado automático. Sin embargo, estas comprobaciones de estado solo están disponibles para instancias integradas en AWS Nitro System.

Activación de las comprobaciones de estado de Amazon EBS para un grupo

Puede activar las comprobaciones de estado de Amazon EBS para grupos de escalado automático nuevos y existentes.

Console

Activación de las comprobaciones de estado de Amazon EBS para un grupo nuevo

Al crear el grupo de escalado automático, en la página Configurar opciones avanzadas, para Comprobaciones de estado, Tipos de comprobaciones de estado adicionales, seleccione Activar las comprobaciones de estado de Amazon EBS. Luego, en Período de gracia de comprobación de estado, ingrese el tiempo, en segundos. Este es la cantidad de tiempo que Amazon EC2 Auto Scaling debe esperar antes de comprobar el estado de una instancia una vez que pasa al estado `InService`. Para obtener más información, consulte Establezca el periodo de gracia de la comprobación de estado para un grupo de escalado automático.

AWS CLI

Activación de las comprobaciones de estado de Amazon EBS para un grupo nuevo

Agregue la opción `--health-check-type` al comando create-auto-scaling-group. En el siguiente ejemplo se especifica **`EBS`** para la opción `--health-check-type` para un nuevo grupo de escalado automático denominado *`my-asg`*.

```
aws autoscaling create-auto-scaling-group --auto-scaling-group-name my-asg \
  --health-check-type "EBS" --health-check-grace-period 60 ...
```

Puede especificar varios valores para la opción `--health-check-type`. Por ejemplo, para añadir los tipos de comprobaciones de estado de Amazon EBS y Elastic Load Balancing, utilice el siguiente comando.

```
aws autoscaling create-auto-scaling-group --auto-scaling-group-name my-asg \
  --health-check-type "EBS,ELB" --health-check-grace-period 60 ...
```

Los nombres de los valores distinguen entre mayúsculas y minúsculas.

Console

Activación de las comprobaciones de estado de Amazon EBS para un grupo existente

1. Abra la consola de Amazon EC2 en https://console.aws.amazon.com/ec2/ y seleccione Auto Scaling Groups en el panel de navegación.
2. En la barra de navegación de la parte superior de la pantalla, elija la misma Región de AWS que utilizó cuando creó el grupo de escalado automático.
3. Seleccione la casilla de verificación situada junto a un grupo existente.

   Se abre un panel dividido en la parte inferior de la página Auto Scaling groups (Grupos de Auto Scaling).
4. En la pestaña Details (Detalles), elija Health checks (Comprobaciones de estado), Edit (Editar).
5. En Comprobaciones de estado, Tipos de comprobaciones de estado adicionales, seleccione Activar las comprobaciones de estado de Amazon EBS.
6. En Período de gracia de comprobación de estado, ingrese el tiempo, en segundos. Este es la cantidad de tiempo que Amazon EC2 Auto Scaling debe esperar antes de comprobar el estado de una instancia una vez que pasa al estado `InService`. Para obtener más información, consulte Establezca el periodo de gracia de la comprobación de estado para un grupo de escalado automático.
7. Elija Actualizar.

AWS CLI

Activación de las comprobaciones de estado de Amazon EBS para un grupo existente

Agregue la opción `--health-check-type` al comando update-auto-scaling-group. El siguiente ejemplo especifica **`EBS`** para la opción `--health-check-type` para un grupo de escalado automático existente denominado *`my-asg`*.

```
aws autoscaling update-auto-scaling-group --auto-scaling-group-name my-asg \
  --health-check-type "EBS" --health-check-grace-period 60
```

Para usar varios tipos de comprobaciones de estado, puede especificar varios valores (por ejemplo, **`EBS,ELB`**) para la opción `--health-check-type`.

Los nombres de los valores distinguen entre mayúsculas y minúsculas.

Desactivación de las comprobaciones de estado de Amazon EBS para un grupo de escalado automático

Desactivación de las comprobaciones de estado de Amazon EBS para un grupo de escalado automático Si ya no necesita las comprobaciones de estado de Amazon EBS, siga el siguiente procedimiento para desactivarlas.

Console

Desactivación de las comprobaciones de estado de Amazon EBS para un grupo

1. Abra la consola de Amazon EC2 en https://console.aws.amazon.com/ec2/ y seleccione Auto Scaling Groups en el panel de navegación.
2. Seleccione la casilla de verificación situada junto a un grupo existente.

   Se abre un panel dividido en la parte inferior de la página Auto Scaling groups (Grupos de Auto Scaling).
3. En la pestaña Details (Detalles), elija Health checks (Comprobaciones de estado), Edit (Editar).
4. En Comprobaciones de estado, Tipos de comprobaciones de estado adicionales, anule la selección de Activar las comprobaciones de estado de Amazon EBS.
5. Elija Actualizar.

AWS CLI

Desactivación de las comprobaciones de estado de Amazon EBS para un grupo

Para actualizar las comprobaciones de estado de un grupo de escalado automático que ya no utilice las comprobaciones de estado de Amazon EBS, utilice el comando update-auto-scaling-group. Incluya la opción `--health-check-type` y un valor de **`EC2`**.

```
aws autoscaling update-auto-scaling-group --auto-scaling-group-name my-asg \
  --health-check-type "EC2"
```

Para desactivar las comprobaciones de estado de Amazon EBS sin desactivar otros tipos de comprobaciones de estado (como Elastic Load Balancing), debe especificarlas en lugar de **`EC2`**. Por ejemplo, para las comprobaciones de estado de Elastic Load Balancing, especifique **`ELB`** para la opción `--health-check-type`.

Los nombres de los valores distinguen entre mayúsculas y minúsculas.

Configuración de una comprobación de estado personalizada para el grupo de escalado automático

Describe cómo integrar las comprobaciones de estado personalizadas con las opciones de comprobación de estado existentes que ofrece Amazon EC2 Auto Scaling. Al combinar las comprobaciones de estado personalizadas con otros tipos de comprobaciones de estado, puede crear un sistema integral de supervisión del estado adaptado a las necesidades de su aplicación.

Para empezar, cree pruebas personalizadas para comprobar que las instancias de su grupo de escalado automático funcionan correctamente y pueden gestionar el tráfico entrante. Si la comprobación de estado que configuró detecta que una instancia no responde, márquela como `Unhealthy`, lo que hará que Amazon EC2 Auto Scaling la sustituya de inmediato.

Puede enviar la información del estado de la instancia directamente a Amazon EC2 Auto Scaling mediante la AWS CLI o un SDK. En los siguientes ejemplos se muestra cómo utilizar el AWS CLI para configurar el estado de una instancia y, a continuación, verificar el estado de la instancia.

Utilice el siguiente comando set-instance-health para establecer el estado de la instancia especificada en **`Unhealthy`**.

```
aws autoscaling set-instance-health --instance-id i-1234567890abcdef0 --health-
status Unhealthy
```

De forma predeterminada, este comando respeta el período de gracia de la comprobación de estado. Sin embargo, puede anular este comportamiento y no respetar el periodo de gracia al incluir la opción `--no-should-respect-grace-period`.

Utilice el comando describe-auto-scaling-groups para verificar que el estado de la instancia sea `Unhealthy`.

```
aws autoscaling describe-auto-scaling-groups --auto-scaling-group-names my-asg
```

A continuación se incluye una respuesta de ejemplo que muestra que el estado de la instancia es `Unhealthy` y que la instancia está terminando.

```
{
```

```
    "AutoScalingGroups": [
        {
            ....
            "Instances": [
                {
                    "ProtectedFromScaleIn": false,
                    "AvailabilityZone": "us-west-2a",
                    "LaunchTemplate": {
                        "LaunchTemplateName": "my-launch-template",
                        "Version": "1",
                        "LaunchTemplateId": "lt-1234567890abcdef0"
                    },
                    "InstanceId": "i-1234567890abcdef0",
                    "InstanceType": "t2.micro",
                    "HealthStatus": "Unhealthy",
                    "LifecycleState": "Terminating"
                },
                ...
            ]
        }
    ]
}
```

Visionado del motivo de los errores de una comprobación de estado

Mediante el siguiente procedimiento, puede ver la información sobre las instancias reemplazadas debido a una comprobación de estado.

De manera predeterminada, Amazon EC2 Auto Scaling crea una nueva actividad de escalado para terminar la instancia en mal estado y, a continuación, la termina. Mientras se termina la instancia, otra actividad de escalado lanza una instancia nueva. Usted puede cambiar este comportamiento para comenzar a inicializar una nueva instancia lo antes posible mediante una política de mantenimiento de instancias. Para obtener más información, consulte Políticas de mantenimiento de instancias.

Console

Visualización del motivo de los fallos de una comprobación de estado

1. Abra la consola de Amazon EC2 en https://console.aws.amazon.com/ec2/ y seleccione Auto Scaling Groups en el panel de navegación.
2. Seleccione la casilla situada junto al grupo de escalado automático.

Se abre un panel dividido en la parte inferior de la página Auto Scaling groups (Grupos de Auto Scaling).

3. En la pestaña Activity (Actividad), en Activity history (Historial de actividad), la columna Status (Estado) muestra si su grupo de escalado automático ha lanzado o terminado las instancias correctamente.

   Si terminó cualquier instancia en mal estado, la columna Cause (Causa) muestra la fecha y la hora de la terminación y el motivo del error de la comprobación de estado. Por ejemplo, `At 2022-05-14T20:11:53Z an instance was taken out of service in response to a user health-check`. Este mensaje indica que una comprobación de estado personalizada marcó la instancia en mal estado.

   Para obtener ayuda con los fallos en las comprobaciones de estado, consulte Solución de problemas de instancias en mal estado en Amazon EC2 Auto Scaling.

AWS CLI

Visualización del motivo de los fallos de una comprobación de estado

Utilice el siguiente comando describe-scaling-activities.

```
aws autoscaling describe-scaling-activities --auto-scaling-group-name my-asg
```

A continuación se incluye una respuesta de ejemplo, donde `Cause` contiene el motivo de fallo en la comprobación de estado.

```
{
  "Activities": [
    {
      "ActivityId": "4c65e23d-a35a-4e7d-b6e4-2eaa8753dc12",
      "AutoScalingGroupName": "my-asg",
      "Description": "Terminating EC2 instance: i-04925c838b6438f14",
      "Cause": "At 2021-04-01T21:48:35Z an instance was taken out of service in
 response to a user health-check.",
      "StartTime": "2021-04-01T21:48:35.859Z",
      "EndTime": "2021-04-01T21:49:18Z",
      "StatusCode": "Successful",
      "Progress": 100,
      "Details": "{\"Subnet ID\":\"subnet-5ea0c127\",\"Availability Zone\":\"us-
west-2a\"...}",
```

```
      "AutoScalingGroupARN": "arn:aws:autoscaling:us-
west-2:123456789012:autoScalingGroup:283179a2-
f3ce-423d-93f6-66bb518232f7:autoScalingGroupName/my-asg"
    },
...
  ]
}
```

Para obtener una descripción de los campos de la salida, consulte Actividad en la Referencia de API de Amazon EC2 Auto Scaling.

Para describir las actividades de escalado después de que se eliminó un grupo de escalado automático, agregue la opción `--include-deleted-groups` al comando describe-scaling-activities.

Utilice las comprobaciones de estado de las aplicaciones con un grupo de Auto Scaling

Al comprobar el estado de las aplicaciones, su grupo de Auto Scaling puede reemplazar las instancias cuyas aplicaciones hayan dejado de responder, incluso cuando la infraestructura subyacente esté en buen estado. Las comprobaciones del estado de las aplicaciones controlan si una aplicación de tu instancia responde a las llamadas HTTP o HTTPS. A diferencia de las comprobaciones de estado de Amazon EC2, que detectan problemas en la instancia o el sistema subyacente, las comprobaciones de estado de las aplicaciones detectan errores en la capa de la aplicación. Por ejemplo, detectan un proceso de servidor web que ha dejado de responder. Si las instancias de su grupo de Auto Scaling tienen comprobaciones de estado de las aplicaciones asociadas e incluidas en la agregación, Amazon EC2 Auto Scaling termina y reemplaza automáticamente las instancias cuyo estado general de la aplicación informa. `impaired` No se requiere ninguna configuración de grupo adicional de Auto Scaling.

Las comprobaciones del estado de las aplicaciones se crean y administran en Amazon EC2, no en Amazon EC2 Auto Scaling. Cree la verificación del estado de la aplicación, asóciela a las instancias del grupo de Auto Scaling por etiqueta y compruebe que los informes son correctos. Para obtener más información sobre la creación de comprobaciones del estado de las aplicaciones, consulte las comprobaciones del estado de las aplicaciones en la guía del usuario de Amazon EC2.

Requisitos previos

Para utilizar las comprobaciones del estado de las aplicaciones con su grupo de Auto Scaling, asegúrese de disponer de lo siguiente:

- Un grupo de Auto Scaling con al menos una instancia.
- Una o más comprobaciones del estado de la aplicación asociadas a las instancias del grupo (mediante etiquetas).
- Un punto final de la aplicación escucha en un puerto de red de cada instancia y responde a las llamadas HTTP o HTTPS en el puerto y la ruta configurados en la comprobación.

Cómo utiliza Amazon EC2 Auto Scaling las comprobaciones de estado de las aplicaciones

Amazon EC2 Auto Scaling usa el estado agregado (general) de la aplicación para cada instancia. Cuando se informa del estado general de la aplicación`impaired`, Amazon EC2 Auto Scaling termina la instancia y lanza una de reemplazo. Esto mantiene la capacidad deseada del grupo.

Amazon EC2 Auto Scaling no actúa en función del estado de una verificación individual. Las comprobaciones marcadas como `excluded` de agregación no afectan al estado general de la aplicación y, por lo tanto, no impulsan las acciones de Amazon EC2 Auto Scaling. Las comprobaciones que se suprimen no impulsan las acciones de Amazon EC2 Auto Scaling.

Amazon EC2 Auto Scaling tampoco actúa sobre otros valores de estado de la aplicación que no sean. `impaired` Si el estado general es`ok`,,, o `initializing` `insufficient-data` `not-applicable``suppressed`, Amazon EC2 Auto Scaling no realiza ninguna acción en función del estado de la aplicación.

Configure las comprobaciones del estado de las aplicaciones para un grupo de Auto Scaling

Para usar las comprobaciones del estado de las aplicaciones con un grupo de Auto Scaling, cree una verificación del estado de la aplicación en Amazon EC2 y, a continuación, asóciela al grupo de Auto Scaling mediante la etiqueta del `aws:autoscaling:groupName` sistema. Amazon EC2 Auto Scaling aplica automáticamente esta etiqueta a todas las instancias del grupo. No es necesario realizar cambios en la plantilla de lanzamiento. Las nuevas instancias se asocian automáticamente cuando se lanzan.

```
aws ec2 associate-application-status-check \
```

```
    --application-status-check-id asc-1234567890abcdef0 \
    --target-tag-associations Key=aws:autoscaling:groupName,Value=my-asg
```

Donde:

- `asc-1234567890abcdef0`— el identificador de la verificación del estado de su solicitud.
- `my-asg`— el nombre de su grupo de Auto Scaling.

Cuando la comprobación se incluye en la agregación, Amazon EC2 Auto Scaling actúa automáticamente en las instancias cuyo estado general de la aplicación informa `impaired` sobre el estado general de la aplicación. No se requiere ninguna configuración de grupo adicional de Auto Scaling.

Para obtener más información sobre la creación y asociación de las comprobaciones de estado de las aplicaciones, consulte las comprobaciones del estado de las aplicaciones en la guía del usuario de Amazon EC2.

Periodo de gracia de la comprobación de estado

Amazon EC2 Auto Scaling utiliza un período de gracia para comprobar el estado para que las nuevas instancias tengan tiempo de iniciarse antes de que comiencen las comprobaciones de estado. Asegúrese de que el período de gracia sea lo suficientemente largo como para que la aplicación comience a responder a las llamadas de verificación de estado una vez que se lance la instancia. Si el período de gracia es demasiado corto, Amazon EC2 Auto Scaling podría terminar y reemplazar las instancias antes de que estén listas para atender el tráfico.

Para obtener más información, consulte Establezca el periodo de gracia de la comprobación de estado para un grupo de escalado automático.

Suprima las comprobaciones del estado de las aplicaciones durante el mantenimiento

Si necesita aplicar parches o realizar tareas de mantenimiento in situ a una instancia de un grupo de Auto Scaling, puede suprimir temporalmente las comprobaciones del estado de las aplicaciones de esa instancia para que Amazon EC2 Auto Scaling no la cancele durante ese período. Para obtener más información sobre la supresión de las comprobaciones de estado de las aplicaciones, consulte la sección sobre la aplicación de In-place parches y el mantenimiento en la guía del usuario de Amazon EC2.

Solución de problemas de instancias en mal estado en Amazon EC2 Auto Scaling

A continuación se muestran los mensajes de error devueltos por Amazon EC2 Auto Scaling, sus posibles causas y los pasos que puede adoptar para resolver los problemas.

Para recuperar un mensaje de error, consulte Visionado del motivo de los errores de una comprobación de estado.

Mensajes de error

- Se quitó del servicio una instancia en respuesta a un error de comprobación del estado de la instancia EC2
- Se quitó del servicio una instancia en respuesta a una comprobación de estado de EC2 que indicaba que se había terminado o detenido
- Se quitó del servicio una instancia en respuesta a un error de comprobación de estado del sistema ELB
- Recursos adicionales

Se quitó del servicio una instancia en respuesta a un error de comprobación del estado de la instancia EC2

Problema: Las instancias de Auto Scaling producen errores en las comprobaciones de estado de Amazon EC2.

Causa 1: si hay problemas que motivan que Amazon EC2 considere que las instancias del grupo de escalado automático están deterioradas, Amazon EC2 Auto Scaling las reemplaza de manera automática como parte de sus comprobaciones de estado.

> **Note**
>
> Amazon EC2 ofrece dos comprobaciones de estado: la comprobación del estado del sistema, que detecta problemas en el host o la infraestructura subyacentes, y la comprobación del estado de la instancia, que detecta problemas con la configuración o el estado de la propia instancia. Este mensaje de error se aplica cuando se produce un error en la comprobación del estado de una instancia o un sistema. Para determinar qué comprobación falló en una instancia específica, consulte los detalles de la

verificación de estado de la instancia en la consola de Amazon EC2 o llame a la `DescribeInstanceStatus` API. También puede usar las CloudWatch métricas de `StatusCheckFailed_Instance` Amazon `StatusCheckFailed_System` y las de Amazon para ver qué comprobación ha fallado. Para obtener más información, consulte las comprobaciones de estado de sus instancias en la guía del usuario de Amazon EC2.

Solución 1: Cuando se produce un error en una comprobación de estado, normalmente debe solucionar el problema usted mismo realizando cambios en la configuración de la instancia hasta que la aplicación deje de presentar problemas. Para resolver este problema, siga estos pasos:

1. Cree manualmente una instancia de Amazon EC2 que no forme parte del grupo de Auto Scaling e investigue el problema. Para obtener ayuda general sobre la investigación de instancias deterioradas, consulte Solución de problemas de las instancias con comprobaciones de estado no superadas en la Guía del usuario de Amazon EC2.
2. Una vez que haya confirmado que su instancia se lanzó correctamente y se encuentra en buen estado, implemente una nueva configuración de instancia sin errores en el grupo de Auto Scaling.
3. Elimine la instancia que ha creado para evitar cargos continuos en la cuenta de AWS .

Se quitó del servicio una instancia en respuesta a una comprobación de estado de EC2 que indicaba que se había terminado o detenido

Problema: Se reemplazan las instancias de Auto Scaling que se han detenido, reiniciado o terminado.

Causa 1: Un usuario detuvo, reinició o terminó manualmente la instancia.

Solución 1: si necesita detener o reiniciar las instancias en su grupo de escalado automático, le recomendamos poner las instancias en espera primero. Para obtener más información, consulte Eliminación temporal de las instancias de un grupo de escalado automático.

Causa 2: Amazon EC2 Auto Scaling intenta reemplazar las instancias de spot después de que el servicio de spot de Amazon EC2 interrumpa las instancias, porque el precio de spot aumenta por encima de su precio máximo o la capacidad ya no está disponible.

Solución 2: No hay garantía de que exista una instancia de spot para cumplir con la solicitud en un momento dado. Sin embargo, puede intentar lo siguiente:

- Utilice un precio máximo de venta al contado más alto (posiblemente el On-Demand precio). Al establecer el precio máximo más alto, ofrece al servicio de spot de Amazon EC2 una mejor oportunidad de lanzar y mantener la cantidad de capacidad requerida.
- Aumente el número de grupos de capacidad diferentes desde los que puede lanzar instancias ejecutando varios tipos de instancias en varias zonas de disponibilidad. Para obtener más información, consulte Grupos de Auto Scaling con varios tipos de instancia y opciones de compra.
- Si utiliza varios tipos de instancias, considere la posibilidad de habilitar la característica de reequilibrio de la capacidad. Esto resulta útil si desea que el servicio de spot de Amazon EC2 intente lanzar una nueva instancia de spot antes de que se termine una instancia en ejecución. Para obtener más información, consulte Reequilibrio de la capacidad en Auto Scaling para reemplazar las instancias de spot en riesgo.

Causa 3: con los bloques de capacidad, Amazon EC2 termina cualquier instancia que siga ejecutándose 30 minutos antes de la hora de finalización del bloque de capacidad. Esta terminación abrupta hace que el grupo de escalado automático intente inicializar instancias nuevas para mantener la capacidad deseada, incluso cuando el bloque de capacidad esté terminando.

Solución 3: para resolver este problema, pruebe lo siguiente:

- Disminuya la capacidad deseada del grupo de escalado automático para evitar que intente inicializar instancias nuevas. Para obtener más información, consulte Escalado manual para Amazon EC2 Auto Scaling.
- Asegúrese de reducir horizontalmente el grupo de escalado automático 30 minutos antes de la hora de finalización del bloque de capacidad para que no aparezca este error con frecuencia. Asegúrese de que todos los enlaces de ciclo de vida se hayan completado 30 minutos antes de la hora de finalización del bloque de capacidad. Para obtener más información, consulte Uso Bloques de capacidad para cargas de trabajo de aprendizaje automático.

Se quitó del servicio una instancia en respuesta a un error de comprobación de estado del sistema ELB

Problema: Las instancias de Auto Scaling pueden pasar las comprobaciones de estado de EC2. Pero pueden no superar las comprobaciones de estado de Elastic Load Balancing para los grupos de destino o los balanceadores de carga clásicos en los que está registrado el grupo de Auto Scaling.

Causa 1: si el grupo de escalado automático se basa en comprobaciones de estado proporcionadas por Elastic Load Balancing, Amazon EC2 Auto Scaling determina el estado mediante la verificación

de los resultados de las comprobaciones de estado de EC2 y de Elastic Load Balancing. El balanceador de carga realiza comprobaciones de estado enviando una solicitud a cada instancia y esperando la respuesta correcta, o estableciendo una conexión con la instancia. Una instancia podría no superar la comprobación de estado de Elastic Load Balancing si una aplicación que se ejecuta en la instancia tiene algún problema como consecuencia del cual el balanceador de carga considera que la instancia se encuentra fuera de servicio.

Solución 1: Para pasar las comprobaciones de estado Elastic Load Balancing:

- Compruebe que la configuración de comprobación de estado de los grupos de destino está configurada correctamente. Puede definir la configuración de comprobación de estado del balanceador de carga por grupo de destino. Para obtener más información, consulte Configuración de las comprobaciones de estado de los destinos.
- Anote los códigos de éxito que el balanceador de carga espera y verifique que la aplicación está configurada correctamente para devolver estos códigos de éxito.
- Compruebe que los grupos de seguridad para el balanceador de carga y el grupo de Auto Scaling están configurados correctamente.
- Compruebe que el balanceador de carga está configurado en las mismas zonas de disponibilidad que el grupo de Auto Scaling.

Solución 2: actualice el grupo de escalado automático para desactivar las comprobaciones de estado de Elastic Load Balancing. Para obtener instrucciones sobre cómo desactivar estas comprobaciones de estado, consulte Desasociar un grupo objetivo o un Equilibrador de carga clásico.

Causa 2: Hay una discrepancia entre el periodo de gracia de la comprobación de estado y el tiempo de inicio de la instancia.

Solución 3: edite el período de gracia de la comprobación de estado para su grupo de escalado automático. Establezca el periodo de gracia en un periodo de tiempo lo suficientemente largo como para admitir el número de comprobaciones de estado consecutivas correctas necesarias antes de que Elastic Load Balancing considere que una instancia recién iniciada está en buen estado. Para obtener más información, consulte Establezca el periodo de gracia de la comprobación de estado para un grupo de escalado automático.

Recursos adicionales

Si tienes otro problema, consulta los siguientes AWS re:Post artículos para obtener ayuda adicional para la solución de problemas:

- ¿Por qué Amazon EC2 Auto Scaling terminó una instancia?
- ¿Por qué Amazon EC2 Auto Scaling no ha terminado una instancia en mal estado?

Panel de estado notificaciones para Amazon EC2 Auto Scaling

Panel de estado Proporciona soporte para las notificaciones que provienen de Amazon EC2 Auto Scaling. Estas notificaciones le ofrecen orientación en sensibilización y corrección de errores para problemas de rendimiento o disponibilidad de recursos que puedan afectar a las aplicaciones. Sólo están disponibles los eventos específicos de grupos de seguridad que faltan y plantillas de lanzamiento.

Panel de estado Es parte del AWS Health servicio. No precisa configuración, y cualquier usuario autenticado en su cuenta puede consultarlo. Para obtener más información, consulte Cómo empezar a utilizar el AWS Health panel de control.

Si recibe un mensaje similar a los siguientes mensajes, debe tratarse como una alarma para tomar medidas.

Ejemplo: El grupo de Auto Scaling no se escala horizontalmente debido a que falta un grupo de seguridad

```
 Hello,

 At 2020-01-11 04:00 UTC, we detected an issue with your Auto Scaling group [ARN] in
Cuenta de AWS 123456789012.

 A security group associated with this Auto Scaling group cannot be found. Each time
a
 scale out operation is performed, it will be prevented until you make a change that
 fixes the issue.

 We recommend that you review and update your Auto Scaling group configuration to
change
 the launch template or launch configuration that depends on the unavailable security
 group.

 Sincerely,
 Amazon Web Services
```

Ejemplo: El grupo de Auto Scaling no se escala horizontalmente debido a que falta una plantilla de lanzamiento

```
 Hello,

 At 2021-05-11 04:00 UTC, we detected an issue with your Auto Scaling group [ARN] in
Cuenta de AWS 123456789012.

 The launch template associated with this Auto Scaling group cannot be found. Each
time
 a scale out operation is performed, it will be prevented until you make a change
that
 fixes the issue.

 We recommend that you review and update your Auto Scaling group configuration and
 specify an existing launch template to use.

 Sincerely,
 Amazon Web Services
```

Supervise CloudWatch las métricas de sus grupos e instancias de Auto Scaling

Las métricas son el concepto fundamental en Amazon CloudWatch. Una métrica representa un conjunto de puntos de datos ordenados por tiempo que se publican en. CloudWatch Una métrica es una variable que hay que monitorizar y los puntos de datos son los valores de esa variable a lo largo del tiempo. Utilice estas métricas para comprobar que el sistema funciona de acuerdo con lo esperado.

Las métricas de Amazon EC2 Auto Scaling que recopilan información acerca de los grupos de escalado automático están en el espacio de nombres de `AWS/AutoScaling`. Las métricas de instancia de Amazon EC2 que recopilan datos de CPU y otros datos de uso de las instancias de escalado automático están en el espacio de nombres de `AWS/EC2`.

La consola de Amazon EC2 Auto Scaling muestra una serie de gráficos para las métricas del grupo y las métricas de instancias agregadas del grupo. Según sus necesidades, es posible que prefiera acceder a los datos de sus grupos e instancias de Auto Scaling desde Amazon CloudWatch en lugar de desde la consola Auto Scaling de Amazon EC2.

Para obtener más información, consulta la Guía del CloudWatch usuario de Amazon.

Contenido

- Visualización de gráficos de supervisión en la consola de Amazon EC2 Auto Scaling
- CloudWatch Métricas de Amazon para Amazon EC2 Auto Scaling
- Configuración de la supervisión para instancias de Auto Scaling

Visualización de gráficos de supervisión en la consola de Amazon EC2 Auto Scaling

En la sección Amazon EC2 Auto Scaling de la consola de Amazon EC2, minute-by-minute puede supervisar el progreso de los grupos individuales de Auto CloudWatch Scaling mediante métricas.

Puede supervisar los siguientes tipos de métricas:

- Métricas de Auto Scaling: las métricas de Auto Scaling se activan solo cuando se habilitan. Para obtener más información, consulte Habilitación de las métricas de grupo de Auto Scaling (consola). Cuando se habilitan las métricas de Auto Scaling, los gráficos de supervisión muestran los datos publicados con una granularidad de un minuto para las métricas de Auto Scaling.
- Métricas de EC2: las métricas de la instancia de Amazon EC2 siempre están habilitadas. Cuando se habilita la supervisión detallada, los gráficos de supervisión muestran los datos publicados con una granularidad de un minuto para las métricas de instancias. Para obtener más información, consulte Configuración de la supervisión para instancias de Auto Scaling.

Para ver los gráficos de supervisión con la consola de Amazon EC2 Auto Scaling

1. Abra la consola Amazon EC2 en https://console.aws.amazon.com/ec2/y seleccione Auto Scaling Groups en el panel de navegación.
2. Seleccione la casilla situada junto al grupo de Auto Scaling para el que desea ver las métricas.

   Se abre un panel dividido en la parte inferior de la página Auto Scaling groups (Grupos de Auto Scaling).
3. Elija la pestaña Monitorización.

   Amazon EC2 Auto Scaling muestra los gráficos de supervisión para las métricas de Auto Scaling.
4. Para ver gráficos de seguimiento de las métricas de instancia agregadas para el grupo, elija EC2.

Acciones sobre los gráficos

- Coloque el cursor sobre un punto de datos para ver una ventana emergente de datos durante una hora específica en UTC.
- Para ampliar un gráfico, elija Enlarge (Ampliar) en la herramienta de menú (los tres puntos verticales) en la parte superior derecha del gráfico. También puede elegir el icono de maximizar en la parte superior del gráfico.
- Ajuste el periodo para los datos que se muestran en el gráfico. Para ello, seleccione uno de los valores predefinidos del periodo. Si se amplía el gráfico, puede elegir la opción Custom (Personalizado) para definir su propio periodo.
- Elija Refresh (Actualizar) en la herramienta de menú para actualizar los datos de un gráfico.
- Arrastre el cursor sobre los datos del gráfico para seleccionar un rango específico. Luego, puede elegir Apply time range (Aplicar intervalo de tiempo) en la herramienta de menú.
- Seleccione Ver registros en la herramienta de menú para ver los flujos de registros asociados (si los hay) en la CloudWatch consola.
- Para ver un gráfico en CloudWatch, selecciona Ver en métricas en la herramienta de menú. Esto te llevará a la CloudWatch página de ese gráfico. Allí, puede ver más información o acceder a información histórica para comprender mejor cómo cambió su grupo de Auto Scaling durante un periodo prolongado.

Métricas de gráficos para los grupos de Auto Scaling

Después de crear un grupo de Auto Scaling, puede abrir la consola de Amazon EC2 Auto Scaling y ver una serie de gráficos de supervisión para el grupo en la pestaña Monitoring (Supervisión).

En la sección Auto Scaling, las métricas de gráficos incluyen las siguientes métricas. Estas métricas proporcionan mediciones que pueden ser indicadores de un posible problema, como la cantidad de instancias en proceso de terminación o la cantidad de instancias pendientes. Puede encontrar definiciones para estas métricas en CloudWatch Métricas de Amazon para Amazon EC2 Auto Scaling.

Display name (Nombre de visualización)	CloudWatch nombre de la métrica
Tamaño mínimo del grupo	`GroupMinSize`

Display name (Nombre de visualización)	CloudWatch nombre de la métrica
Tamaño máximo del grupo	`GroupMaxSize`
Capacidad deseada	`GroupDesiredCapacity`
Instancias en servicio	`GroupInServiceInstances`
Instancias pendientes	`GroupPendingInstances`
Instancias en espera	`GroupStandbyInstances`
Instancias en proceso de terminación	`GroupTerminatingInstances`
Total de instancias	`GroupTotalInstances`

En la sección EC2, puede encontrar las siguientes métricas de gráficos basadas en las métricas de rendimiento clave de sus instancias de Amazon EC2. Estas métricas de EC2 son una agrupación de las métricas de todas las instancias del grupo. Puede encontrar las definiciones de estas métricas en Enumere las CloudWatch métricas disponibles para sus instancias en la Guía del usuario de Amazon EC2.

Display name (Nombre de visualización)	CloudWatch nombre de la métrica
Utilización de la CPU	`CPUUtilization`
Lecturas en disco	`DiskReadBytes`
Operaciones de lectura en disco	`DiskReadOps`
Escrituras en disco	`DiskWriteBytes`
Operaciones de escritura en disco	`DiskWriteOps`

Display name (Nombre de visualización)	CloudWatch nombre de la métrica
Entrada de red	`NetworkIn`
Salida de red	`NetworkOut`
Comprobación de estado no superada (cualquiera)	`StatusCheckFailed`
Comprobación de estado no superada (instancia)	`StatusCheckFailed_Instance`
Comprobación de estado no superada (sistema)	`StatusCheckFailed_System`

Además, algunas métricas están disponibles para casos de uso específicos en las métricas gráficas de Auto Scaling.

Las siguientes métricas gráficas están disponibles para grupos en los que las instancias tienen ponderaciones que definen con cuántas unidades contribuye cada instancia a la capacidad deseada del grupo. Puede encontrar definiciones para estas métricas en CloudWatch Métricas de Amazon para Amazon EC2 Auto Scaling.

Display name (Nombre de visualización)	CloudWatch nombre de la métrica
Unidades de capacidad en servicio	`GroupInServiceCapacity`
Unidades de capacidad pendientes	`GroupPendingCapacity`
Unidades de capacidad en espera	`GroupStandbyCapacity`
Unidades de capacidad en proceso de terminación	`GroupTerminatingCapacity`

Display name (Nombre de visualización)	CloudWatch nombre de la métrica
Unidades de capacidad total	`GroupTotalCapacity`

Las siguientes métricas son útiles si su grupo usa la característica de grupo en caliente. Puede encontrar definiciones para estas métricas en CloudWatch Métricas de Amazon para Amazon EC2 Auto Scaling.

Display name (Nombre de visualización)	CloudWatch nombre de la métrica
Tamaño mínimo de grupo en caliente	`WarmPoolMinSize`
Capacidad deseada de grupo en caliente	`WarmPoolDesiredCapacity`
Unidades de capacidad pendientes de grupo en caliente	`WarmPoolPendingCapacity`
Unidades de capacidad en proceso de terminación de grupo en caliente	`WarmPoolTerminatingCapacity`
Unidades de capacidad calentadas de grupo en caliente	`WarmPoolWarmedCapacity`
Lanzamiento de unidades de capacidad total de grupo en caliente	`WarmPoolTotalCapacity`
Capacidad deseada de grupo y grupo en caliente	`GroupAndWarmPoolDesiredCapacity`

Display name (Nombre de visualización)	CloudWatch nombre de la métrica
Lanzamiento de unidades de capacidad total de grupo y grupo en caliente	`GroupAndWarmPoolTotalCapacity`

Recursos relacionados

- Para supervisar las métricas por instancia, consulte Representación gráfica de métricas para las instancias en la Guía del usuario de Amazon EC2.
- CloudWatch los paneles son páginas de inicio personalizables en la CloudWatch consola. Puede utilizar estas páginas para monitorizar sus recursos en una sola vista, incluidos los recursos que están distribuidos en diferentes regiones. Puede usar los CloudWatch paneles para crear vistas personalizadas de las métricas y las alarmas de sus AWS recursos. Para obtener más información, consulta la Guía del CloudWatch usuario de Amazon.

CloudWatch Métricas de Amazon para Amazon EC2 Auto Scaling

Amazon EC2 Auto Scaling publica las siguientes métricas en el espacio de nombres de `AWS/AutoScaling`. Las métricas de grupo de escalado automático que estén disponibles dependerán de si tiene habilitadas las métricas de grupo y de las métricas de grupo que haya activado. Las métricas están disponibles con una granularidad de un minuto sin cargo adicional, pero debe habilitarlas.

Al activar las métricas de grupo de Auto Scaling, Amazon EC2 Auto Scaling envía los datos muestreados CloudWatch a cada minuto en función del mejor esfuerzo. En raras ocasiones, cuando se produce una CloudWatch interrupción en el servicio, los datos no se rellenan para cubrir los vacíos en el historial de métricas del grupo.

Contenido

- Métricas de grupo de Auto Scaling
- Dimensiones de las métricas del grupo de Auto Scaling
- Dimensiones y métricas de escalado predictivo
- Habilitación de las métricas de grupo de Auto Scaling (consola)
- Habilitación de las métricas de grupo de Auto Scaling (AWS CLI)

Métricas de grupo de Auto Scaling

Con estas métricas, obtendrá una visibilidad prácticamente continua del historial de su grupo de escalado automático, como los cambios en el tamaño del grupo a lo largo del tiempo.

Métrica	Description (Descripción)
`GroupMinSize`	El tamaño mínimo del grupo de Auto Scaling. Criterios del informe: se notifica si la recopilación de métricas está habilitada.
`GroupMaxSize`	El tamaño máximo del grupo de Auto Scaling. Criterios del informe: se notifica si la recopilación de métricas está habilitada.
`GroupDesiredCapacity`	El número de instancias que el grupo de Auto Scaling intenta mantener. Criterios del informe: se notifica si la recopilación de métricas está habilitada.
`GroupInServiceInstances`	El número de instancias que se ejecutan como parte del grupo de Auto Scaling. Esta métrica no incluye las instancias que están pendientes o se están terminando. Criterios del informe: se notifica si la recopilación de métricas está habilitada.
`GroupPendingInstances`	El número de instancias que están pendientes. Una instancia pendiente aún no está operativa. Esta métrica no incluye las instancias que están en servicio o se están terminando. Criterios del informe: se notifica si la recopilación de métricas está habilitada.
`GroupStandbyInstances`	El número de instancias que tienen el estado `Standby`. Las instancias con este estado se siguen ejecutando pero no están en servicio.

Métrica	Description (Descripción)
	Criterios del informe: se notifica si la recopilación de métricas está habilitada.
`GroupTerminatingInstances`	El número de instancias que se están terminando. Esta métrica no incluye las instancias que están en servicio, pendientes o que regresan a un grupo preparado después de la reducción horizontal del grupo de escalado automático. Criterios del informe: se notifica si la recopilación de métricas está habilitada.
`GroupTerminatingRetainedInstances`	El número de instancias en el `Terminating:Retained` estado. Criterios del informe: se notifica si la recopilación de métricas está habilitada.
`GroupTotalInstances`	El número total de instancias en el grupo de Auto Scaling. Esta métrica identifica el número de instancias que están en servicio, pendientes y en proceso de terminación. Criterios del informe: se notifica si la recopilación de métricas está habilitada.

Cuando configura un grupo de instancias mixtas para medir la capacidad deseada en diferentes unidades, por ejemplo, asignando pesos en función del recuento de vCPU de cada tipo de instancia, las siguientes métricas cuentan la cantidad de unidades que usa su grupo de escalado automático. Si no configuró un grupo de instancias mixtas para medir la capacidad deseada en diferentes unidades, se rellenan las siguientes métricas, pero son iguales a las métricas que se definen en la tabla anterior. Para obtener más información, consulte Información general de la configuración de creación de un grupo de instancias mixtas.

Métrica	Description (Descripción)
`GroupInServiceCapacity`	El número de unidades de capacidad que se ejecutan como parte del grupo de Auto Scaling.

Métrica	Description (Descripción)
	Criterios del informe: se notifica si la recopilación de métricas está habilitada.
`GroupPendingCapacity`	El número de unidades de capacidad que están pendientes. Criterios del informe: se notifica si la recopilación de métricas está habilitada.
`GroupStandbyCapacity`	El número de unidades de capacidad que están en un estado `Standby`. Criterios del informe: se notifica si la recopilación de métricas está habilitada.
`GroupTerminatingCapacity`	El número de unidades de capacidad que están en proceso de terminación. Criterios del informe: se notifica si la recopilación de métricas está habilitada.
`GroupTerminatingRetainedCapacity`	Las unidades de capacidad representadas por instancias en el `Terminating:Retained` estado. Criterios del informe: se notifica si la recopilación de métricas está habilitada.
`GroupTotalCapacity`	El número total de unidades de capacidad del grupo de Auto Scaling. Criterios del informe: se notifica si la recopilación de métricas está habilitada.

Amazon EC2 Auto Scaling también informa de las siguientes métricas para los grupos de escalado automático que tienen grupo de calentamiento. Para obtener más información, consulte Reducción de la latencia de las aplicaciones con tiempos de arranque prolongados mediante grupos de calentamiento.

Métrica	Description (Descripción)
`WarmPoolMinSize`	Tamaño mínimo del grupo de calentamiento. Criterios del informe: se notifica si la recopilación de métricas está habilitada.
`WarmPoolDesiredCapacity`	Cantidad de capacidad que Amazon EC2 Auto Scaling intenta mantener en el grupo de calentamiento. Esto equivale al tamaño máximo del grupo de Auto Scaling menos la capacidad deseada o, si se establece, a la capacidad máxima preparada del grupo de Auto Scaling menos la capacidad deseada. Sin embargo, cuando el tamaño mínimo del grupo de calentamiento es igual o superior a la diferencia entre el tamaño máximo (o, si se establece, la capacidad máxima preparada) y la capacidad deseada del grupo de Auto Scaling, la capacidad deseada del grupo de calentamiento será equivalente a `WarmPoolMinSize` . Criterios del informe: se notifica si la recopilación de métricas está habilitada.
`WarmPoolPendingCapacity`	Cantidad de capacidad del grupo de calentamiento que está pendiente. Esto incluye las instancias que regresan a un grupo preparado después de que el grupo de escalado automático se reduzca horizontalmente. Esta métrica no incluye las instancias en ejecución, detenidas, o en proceso de terminación. Criterios del informe: se notifica si la recopilación de métricas está habilitada.
`WarmPoolPendingRetainedCapacity`	Rastrea las instancias retenidas que regresaron a la piscina caliente. Criterios del informe: se notifica si la recopilación de métricas está habilitada.

Métrica	Description (Descripción)
`WarmPoolTerminatingCapacity`	Cantidad de capacidad del grupo de calentamiento que está en proceso de terminación. Esta métrica no incluye las instancias en ejecución, detenidas o pendientes. Criterios del informe: se notifica si la recopilación de métricas está habilitada.
`WarmPoolTerminatingRetainedCapacity`	Las pistas retenían las instancias que terminaban en la piscina caliente. Criterios del informe: se notifica si la recopilación de métricas está habilitada.
`WarmPoolWarmedCapacity`	Cantidad de capacidad disponible para ingresar al grupo de Auto Scaling durante el escalado horizontal. Esta métrica no incluye las instancias que están pendientes o se están terminando. Criterios del informe: se notifica si la recopilación de métricas está habilitada.
`WarmPoolTotalCapacity`	Capacidad total del grupo de calentamiento, incluidas las instancias en ejecución, detenidas, pendientes o en proceso de terminación. Criterios del informe: se notifica si la recopilación de métricas está habilitada.
`GroupAndWarmPoolDesiredCapacity`	Capacidad deseada del grupo de Auto Scaling y del grupo de calentamiento combinadas. Criterios del informe: se notifica si la recopilación de métricas está habilitada.

Métrica	Description (Descripción)
`GroupAndWarmPoolTotalCapacity`	Capacidad total del grupo de Auto Scaling y del grupo de calentamiento combinadas. Esto incluye instancias en ejecución , detenidas, pendientes, en proceso de terminación o en servicio. Criterios del informe: se notifica si la recopilación de métricas está habilitada.

Dimensiones de las métricas del grupo de Auto Scaling

Puede utilizar las siguientes dimensiones para ajustar las métricas mostradas en las tablas anteriores.

Dimensión	Description (Descripción)
`AutoScalingGroupName`	Filtra según el nombre de un grupo de Auto Scaling.

Dimensiones y métricas de escalado predictivo

El espacio de nombres de `AWS/AutoScaling` incluye las siguientes métricas para el escalado predictivo.

Las métricas están disponibles con una resolución de una hora.

Puede evaluar la precisión de las previsiones comparando los valores previstos con los valores reales. Para obtener más información acerca de la evaluación de la precisión del pronóstico mediante estas métricas, consulte Supervise las métricas de escalado predictivo con CloudWatch.

Métrica	Description (Descripción)	Dimensiones
`PredictiveScalingLoadForecast`	La cantidad de carga que se prevé que generará su aplicación. Las estadísticas `Average`, `Minimum`, y `Maximum` son útiles, pero la estadística `Sum` no lo es.	`AutoScalingGroupName` , `PolicyName` , `PairIndex`

Métrica	Description (Descripción)	Dimensiones
	Reporting criteria (Criterios de informes): se informa después de crear la previsión inicial.	
`PredictiveScalingCapacityForecast`	La cantidad anticipada de capacidad necesaria para satisfacer la demanda de las aplicaciones. Esto se basa en la previsión de carga y el nivel de utilización objetivo en el que desea mantener las instancias de escalado automático. Las estadísticas `Average`, `Minimum`, y `Maximum` son útiles, pero la estadística `Sum` no lo es. Reporting criteria (Criterios de informes): se informa después de crear la previsión inicial.	`AutoScalingGroupName`, `PolicyName`
`PredictiveScalingMetricPairCorrelation`	La correlación entre la métrica de escalado y el promedio por instancia de la métrica de carga. El escalado predictivo supone una alta correlación. Por lo tanto, si observa un valor bajo en esta métrica, es mejor no usar un par de métricas. Las estadísticas `Average`, `Minimum`, y `Maximum` son útiles, pero la estadística `Sum` no lo es. Reporting criteria (Criterios de informes): se informa después de crear la previsión inicial.	`AutoScalingGroupName`, `PolicyName`, `PairIndex`

> **Note**
>
> La dimensión `PairIndex` devuelve información asociada al índice del par de métricas de escalado de carga asignado por Amazon EC2 Auto Scaling. El único valor válido actualmente es `0`.

Habilitación de las métricas de grupo de Auto Scaling (consola)

Para habilitar las métricas de grupo

1. Abra la consola Amazon EC2 en https://console.aws.amazon.com/ec2/y seleccione Auto Scaling Groups en el panel de navegación.
2. Seleccione la casilla situada junto al grupo de escalado automático.

   Se abre un panel dividido en la parte inferior de la página.
3. En la página Monitoring (Monitoreo), seleccione la colección de métricas del grupo de Auto Scaling, y la casilla Enable (Habilitar) que se encuentra en la parte superior de la página en Auto Scaling.

Para desactivar las métricas de grupo

1. Abra la consola Amazon EC2 en https://console.aws.amazon.com/ec2/y seleccione Auto Scaling Groups en el panel de navegación.
2. Seleccione el grupo de escalado automático.
3. En la pestaña Monitoring (Monitoreo), desmarque la casilla Auto Scaling group metrics collection (Colección de métricas de grupo de Auto Scaling), Enable (Habilitar).

Habilitación de las métricas de grupo de Auto Scaling (AWS CLI)

Habilitación de las métricas de grupo de escalado automático

Active una o más métricas de grupo mediante el enable-metrics-collectioncomando. Por ejemplo, el comando siguiente habilita una única métrica para el grupo de escalado automático especificado.

```
aws autoscaling enable-metrics-collection --auto-scaling-group-name my-asg \
  --metrics GroupDesiredCapacity --granularity "1Minute"
```

Si omite la opción `--metrics`, se habilitan todas las métricas.

```
aws autoscaling enable-metrics-collection --auto-scaling-group-name my-asg \
  --granularity "1Minute"
```

Deshabilitación de las métricas de grupo de escalado automático

Utilice el disable-metrics-collectioncomando para deshabilitar todas las métricas de grupo.

```
aws autoscaling disable-metrics-collection --auto-scaling-group-name my-asg
```

Configuración de la supervisión para instancias de Auto Scaling

Amazon EC2 recopila y procesa los datos sin procesar de las instancias, y los convierte en métricas legibles prácticamente en tiempo real que describen la CPU y otros datos de uso de su grupo de escalado automático. Puede configurar el intervalo para supervisar estas métricas eligiendo la granularidad de uno o cinco minutos.

La supervisión de instancias se habilita cada vez que se lanza una, ya sea la supervisión básica (granularidad de cinco minutos) o la supervisión detallada (granularidad un minuto). Para la monitorización detallada, se aplican cargos adicionales. Para obtener más información, consulte CloudWatch los precios de Amazon y la supervisión de sus instancias CloudWatch en la Guía del usuario de Amazon EC2.

Antes de crear un grupo de escalado automático, debe crear una configuración o una plantilla de lanzamiento que permita el tipo de supervisión que sea adecuado para su aplicación. Si agrega una política de escalamiento a su grupo, le recomendamos encarecidamente que utilice la supervisión detallada a fin de obtener datos de las métricas para las instancias de EC2 a una granularidad de un minuto, ya que esto permite una respuesta más rápida a los cambios en la carga.

Contenido

- Habilitación del monitoreo detallado (consola)
- Habilitar el monitoreo detallado (AWS CLI)
- Cambio entre la supervisión básica y detallada
- Recopile métricas adicionales con el CloudWatch agente

Habilitación del monitoreo detallado (consola)

De forma predeterminada, la supervisión básica está habilitada cuando se utiliza Consola de administración de AWS para crear una plantilla de lanzamiento o una configuración de lanzamiento.

Para habilitar el monitoreo detallado en una plantilla de lanzamiento

Al crear la plantilla de lanzamiento mediante la Consola de administración de AWS opción CloudWatchSupervisión detallada de la sección Detalles avanzados, seleccione Activar. De lo contrario, se habilita la monitorización básica. Para obtener más información, consulte Creación de una plantilla de lanzamiento mediante la configuración avanzada.

Para habilitar el monitoreo detallado en una configuración de lanzamiento

Al crear la configuración de lanzamiento mediante la Consola de administración de AWS, en la sección Configuración adicional, seleccione Habilitar la supervisión detallada de la instancia EC2. CloudWatch De lo contrario, se habilita la monitorización básica. Para obtener más información, consulte Crear una configuración de lanzamiento.

Habilitar el monitoreo detallado (AWS CLI)

De forma predeterminada, el monitoreo básico se habilita al crear una plantilla de lanzamiento mediante la AWS CLI. El monitoreo detallado se habilita de forma predeterminada cuando crea una configuración de lanzamiento mediante la AWS CLI.

Para habilitar el monitoreo detallado en una plantilla de lanzamiento

Para las plantillas de lanzamiento, utilice el comando create-launch-template y transfiera un archivo JSON que contenga la información para crear la plantilla de lanzamiento. Establezca el atributo de monitorización en `"Monitoring":{"Enabled":true}` para habilitar la monitorización detallada o en `"Monitoring":{"Enabled":false}` para habilitar la monitorización básica.

Para habilitar el monitoreo detallado en una configuración de lanzamiento

Para las configuraciones de lanzamiento, utilice el comando create-launch-configuration con la opción `--instance-monitoring`. Establezca esta opción en `true` para habilitar la monitorización detallada o en `false` para habilitar la monitorización básica.

```
--instance-monitoring Enabled=true
```

Cambio entre la supervisión básica y detallada

Para cambiar el tipo de monitoreo habilitado en las instancias EC2 nuevas, actualice la plantilla de lanzamiento o el grupo de Auto Scaling para que utilicen una nueva plantilla de lanzamiento o configuración de lanzamiento. Las instancias existentes siguen utilizando el tipo de monitorización que estaba habilitado anteriormente. Para actualizar todas las instancias, termínelas de forma que se sustituyan por el grupo de Auto Scaling o actualice las instancias individualmente mediante monitor-instances y unmonitor-instances.

> **Note**
>
> Con las características de duración máxima de la instancia y actualización de instancias, también puede reemplazar todas las instancias en el grupo de Auto Scaling para lanzar nuevas instancias que utilicen la nueva configuración. Para obtener más información, consulte Reemplazo de las instancias del grupo de escalado automático.

Al cambiar entre el monitoreo básico y detallado:

Si tiene CloudWatch alarmas asociadas a las políticas de escalado por pasos o políticas de escalado simples para su grupo de Auto Scaling, utilice el put-metric-alarmcomando para actualizar cada alarma. Ajuste cada periodo para que coincida con el tipo de monitorización (300 segundos para la monitorización básica y 60 segundos para la monitorización detallada). Si cambia de la monitorización detallada a la monitorización básica, pero no actualiza las alarmas para que coincidan con el periodo de cinco minutos, se siguen comprobando las estadísticas cada minuto. Es posible que no haya datos disponibles durante cuatro de cada cinco periodos.

Recopile métricas adicionales con el CloudWatch agente

Para recopilar métricas a nivel del sistema operativo, como la memoria disponible y utilizada, debe instalar el CloudWatch agente. Pueden aplicarse cargos adicionales. Puede usar el CloudWatch agente para recopilar tanto las métricas del sistema como los archivos de registro de las instancias de Amazon EC2. Para obtener más información, consulta las métricas recopiladas por el CloudWatch agente en la Guía del CloudWatch usuario de Amazon.

Registre las llamadas a la API Amazon EC2 Auto Scaling con AWS CloudTrail

Amazon EC2 Auto Scaling está integrado AWS CloudTrailcon un servicio que proporciona un registro de las acciones realizadas por un usuario, rol o Servicio de AWS un. CloudTrail captura las llamadas a la API para Auto Scaling como eventos. Las llamadas capturadas incluyen llamadas desde la Consola de administración de AWS y llamadas de código a las operaciones de la API de escalado automático. Con la información recopilada por CloudTrail, puede determinar la solicitud que se realizó a Auto Scaling, la dirección IP desde la que se realizó la solicitud, cuándo se realizó y detalles adicionales.

Cada entrada de registro o evento contiene información sobre quién generó la solicitud. La información de identidad del usuario le ayuda a determinar lo siguiente:

- Si la solicitud se realizó con las credenciales del usuario raíz o del usuario.
- Si la solicitud se realizó en nombre de un usuario de IAM Identity Center.
- Si la solicitud se realizó con credenciales de seguridad temporales de un rol o fue un usuario federado.
- Si la solicitud la realizó otro Servicio de AWS.

CloudTrail está activa en tu cuenta Cuenta de AWS cuando creas la cuenta y tienes acceso automáticamente al historial de CloudTrail eventos. El historial de CloudTrail eventos proporciona un registro visible, consultable, descargable e inmutable de los últimos 90 días de eventos de gestión registrados en un. Región de AWSPara obtener más información, consulte Uso del historial de CloudTrail eventos en la Guía del usuario.AWS CloudTrail La visualización del historial de eventos no conlleva ningún CloudTrail cargo.

Para tener un registro continuo de los eventos de Cuenta de AWS los últimos 90 días, crea una ruta.

CloudTrail senderos

Un rastro permite CloudTrail entregar archivos de registro a un bucket de Amazon S3. Todos los senderos creados con él Consola de administración de AWS son multirregionales. Puede crear un registro de seguimiento de una sola región o multirregionales mediante la AWS CLI. Se recomienda crear un sendero multirregional, ya que puedes capturar toda la actividad de tu Regiones de AWS cuenta. Si crea un registro de seguimiento de una sola región, solo podrá ver los eventos registrados en la Región de AWS del registro de seguimiento. Para obtener

más información acerca de los registros de seguimiento, consulte Creación de un registro de seguimiento para su Cuenta de AWS y Creación de un registro de seguimiento para una organización en la Guía del usuario de AWS CloudTrail .

Puede enviar una copia de sus eventos de administración en curso a su bucket de Amazon S3 sin coste alguno CloudTrail mediante la creación de una ruta; sin embargo, hay cargos por almacenamiento en Amazon S3. Para obtener más información sobre CloudTrail los precios, consulte AWS CloudTrail Precios. Para obtener información acerca de los precios de Amazon S3, consulte Precios de Amazon S3.

Eventos de administración de Auto Scaling en CloudTrail

Los eventos de administración proporcionan información sobre las operaciones de administración que se realizan en los recursos de su empresa Cuenta de AWS. Se denominan también operaciones del plano de control. De forma predeterminada, CloudTrail registra los eventos de administración.

Amazon EC2 Auto Scaling registra todas las operaciones de plano de control de escalado automático como eventos de administración. Para obtener una lista de las operaciones del plano de control de Amazon EC2 Auto Scaling en las que Auto Scaling registra CloudTrail, consulte la referencia de la API Amazon EC2 Auto Scaling.

Ejemplos de eventos de escalado automático

Un evento representa una solicitud única de cualquier fuente e incluye información sobre la operación de API solicitada, la fecha y la hora de la operación, los parámetros de la solicitud, etc. CloudTrail Los archivos de registro no son un registro ordenado de las llamadas a la API pública, por lo que los eventos no aparecen en ningún orden específico.

En el siguiente ejemplo, se muestra un CloudTrail evento que demuestra la `CreateLaunchConfiguration` operación.

```
{
    "eventVersion": "1.05",
    "userIdentity": {
        "type": "Root",
        "principalId": "123456789012",
        "arn": "arn:aws:iam::123456789012:root",
        "accountId": "123456789012",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "sessionContext": {
```

```
            "attributes": {
                "mfaAuthenticated": "false",
                "creationDate": "2018-08-21T17:05:42Z"
            }
        }
    },
    "eventTime": "2018-08-21T17:07:49Z",
    "eventSource": "autoscaling.amazonaws.com",
    "eventName": "CreateLaunchConfiguration",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "Coral/Jakarta",
    "requestParameters": {
        "ebsOptimized": false,
        "instanceMonitoring": {
            "enabled": false
        },
        "instanceType": "t2.micro",
        "keyName": "EC2-key-pair-oregon",
        "blockDeviceMappings": [
            {
                "deviceName": "/dev/xvda",
                "ebs": {
                    "deleteOnTermination": true,
                    "volumeSize": 8,
                    "snapshotId": "snap-01676e0a2c3c7de9e",
                    "volumeType": "gp2"
                }
            }
        ],
        "launchConfigurationName": "launch_configuration_1",
        "imageId": "ami-6cd6f714d79675a5",
        "securityGroups": [
            "sg-00c429965fd921483"
        ]
    },
    "responseElements": null,
    "requestID": "0737e2ea-fb2d-11e3-bfd8-99133058e7bb",
    "eventID": "3fcfb182-98f8-4744-bd45-b38835ab61cb",
    "eventType": "AwsApiCall",
    "recipientAccountId": "123456789012"
}
```

Para obtener información sobre el contenido de los CloudTrail registros, consulte el contenido de los CloudTrail registros en la Guía del AWS CloudTrail usuario.

Auto RemoveAction Scaling activa CloudWatch

Su AWS CloudTrail registro puede mostrar que Auto Scaling llama a la CloudWatch `RemoveAction` API cuando Auto Scaling le indica CloudWatch que elimine la acción de escalado automático de una alarma. Esto puede suceder si anula el registro de un objetivo escalable, elimina una política de escalado o si una alarma invoca una política de escalado inexistente.

Opciones de notificaciones de Amazon SNS para Amazon EC2 Auto Scaling

Puede configurar su grupo de escalado automático para que le informe eventos importantes que afecten a su aplicación. Con notificaciones, también puede eliminar los sondeos de modo que no encuentre el error `RequestLimitExceeded` que a veces se produce con el sondeo.

Hay dos formas de recibir notificaciones sobre Amazon EC2 Auto Scaling:

- Amazon Simple Notification Service: Amazon SNS puede enviarle una notificación cada vez que el grupo de escalado automático inicialice o termine instancias. Solo puede activar o desactivar notificaciones de Amazon SNS. Para obtener más información, consulte Amazon SNS y Amazon EC2 Auto Scaling.
- Amazon EventBridge: EventBridge proporciona notificaciones más avanzadas basadas en eventos que coinciden con criterios específicos y se envían a una variedad de destinos, incluido Amazon SNS. EventBridge también puede monitorear una gama más amplia de eventos de Auto Scaling para un monitoreo más preciso. Para obtener más información, consulte Se usa EventBridge para gestionar eventos de Auto Scaling.

Si lo desea, puede utilizar notificaciones con enlaces de ciclo de vida para realizar acciones personalizadas en las instancias durante la inicialización o la finalización. Para obtener más información sobre cómo configurar las notificaciones para usarlas con los enlaces del ciclo de vida, consulte Enlaces de ciclo de vida de Amazon EC2 Auto Scaling.

Amazon SNS y Amazon EC2 Auto Scaling

En esta sección se muestra cómo usar Amazon SNS para supervisar cuándo su grupo de escalado automático inicializa o termina instancias.

Por ejemplo, si configura el grupo de Auto Scaling para que utilice el tipo de notificación `autoscaling: EC2_INSTANCE_TERMINATE`, y su grupo de Auto Scaling termina una instancia, se envía una notificación por correo electrónico. Este correo electrónico contiene los detalles de la instancia terminada, como el ID de instancia y el motivo por el que se terminó la instancia.

Tenga en cuenta que, a medida que Amazon EC2 Auto Scaling agrega o elimina instancias del grupo, se le envían notificaciones sobre estos cambios, con una notificación por instancia. Sin embargo, la entrega de estas notificaciones se realiza en la medida de lo posible, y sus instancias podrían fallar después de la notificación inicial, por ejemplo, si falla una comprobación de estado posterior. Para obtener más información sobre el proceso de comprobaciones de estado, consulte Comprobaciones de estado para instancias en un grupo de escalado automático.

Para obtener más información sobre Amazon SNS en general, consulte la Guía para desarrolladores de Amazon Simple Notification Service.

Contenido

- Notificaciones de SNS
- Configuración de notificaciones de Amazon SNS para Amazon EC2 Auto Scaling
  - Crear un tema de Amazon SNS
  - Suscripción al tema de Amazon SNS
  - Confirmación de la suscripción a Amazon SNS
  - Configuración de un grupo de Auto Scaling para enviar notificaciones
  - Prueba de la notificación
  - Eliminación de la configuración de notificaciones
- Política de claves para un tema de Amazon SNS cifrado

Notificaciones de SNS

Amazon EC2 Auto Scaling admite el envío de notificaciones de Amazon SNS cuando se producen los eventos siguientes.

Event	Description (Descripción)
`autoscaling:EC2_INSTANCE_LAUNCH`	Instancia lanzada correctamente

Event	Description (Descripción)
`autoscaling:EC2_INSTANCE_LAUNCH_ERROR`	Error al lanzar la instancia
`autoscaling:EC2_INSTANCE_TERMINATE`	Instancia terminada correctamente
`autoscaling:EC2_INSTANCE_TERMINATE_ERROR`	Error al terminar la instancia

El mensaje incluye la siguiente información:

- `Event`: el evento.
- `AccountId`: el ID de la cuenta de Amazon Web Services.
- `AutoScalingGroupName`: el nombre del grupo de Auto Scaling.
- `AutoScalingGroupARN`: el ARN del grupo de Auto Scaling.
- `EC2InstanceId`: el ID de la instancia EC2.

Por ejemplo:

```
Service: AWS Auto Scaling
Time: 2016-09-30T19:00:36.414Z
RequestId: 4e6156f4-a9e2-4bda-a7fd-33f2ae528958
Event: autoscaling:EC2_INSTANCE_LAUNCH
AccountId: 123456789012
AutoScalingGroupName: my-asg
AutoScalingGroupARN: arn:aws:autoscaling:region:123456789012:autoScalingGroup...
ActivityId: 4e6156f4-a9e2-4bda-a7fd-33f2ae528958
Description: Launching a new EC2 instance: i-0598c7d356eba48d7
Cause: At 2016-09-30T18:59:38Z a user request update of AutoScalingGroup constraints
 to ...
StartTime: 2016-09-30T19:00:04.445Z
EndTime: 2016-09-30T19:00:36.414Z
StatusCode: InProgress
StatusMessage:
Progress: 50
EC2InstanceId: i-0598c7d356eba48d7
Details: {"Subnet ID":"subnet-id","Availability Zone":"zone"}
```

```
Origin: AutoScalingGroup
Destination: EC2
```

Configuración de notificaciones de Amazon SNS para Amazon EC2 Auto Scaling

Para utilizar Amazon SNS para enviar notificaciones por correo electrónico, primero debe crear un tema y, a continuación, suscribir sus direcciones de correo electrónico al tema.

Crear un tema de Amazon SNS

Un tema de SNS es un punto de acceso lógico, un canal de comunicación que el grupo de Auto Scaling utiliza para enviar las notificaciones. Los temas se crean especificando un nombre para el tema.

Los nombres de tema creados deben cumplir los siguientes requisitos:

- Deben tener entre 1 y 256 caracteres.
- Deben contener letras ASCII en mayúsculas y minúsculas, números, guiones bajos o guiones.

Para obtener instrucciones, consulte el tema Creación de un tema de Amazon SNS en la Guía para desarrolladores de Amazon Simple Notification Service.

Suscripción al tema de Amazon SNS

Para recibir las notificaciones que su grupo de escalado automático envía al tema, debe suscribir un punto de conexión al tema. En este procedimiento, en Punto de conexión, especifique la dirección de correo electrónico donde desea recibir las notificaciones de Amazon EC2 Auto Scaling.

Para obtener más información, consulte el tema Suscripción a un tema de Amazon SNS en la Guía para desarrolladores de Amazon Simple Notification Service.

Confirmación de la suscripción a Amazon SNS

Amazon SNS envía un correo electrónico de confirmación a la dirección de correo electrónico que ha especificado en el paso anterior.

Asegúrese de abrir el correo electrónico de las notificaciones de AWS y de elegir el enlace para confirmar la suscripción antes de continuar en el siguiente paso.

Recibirá un mensaje de confirmación de AWS. Amazon SNS estará ahora configurado para recibir notificaciones y enviar la notificación como un email a la dirección especificada.

Configuración de un grupo de Auto Scaling para enviar notificaciones

Puede configurar su grupo de Auto Scaling para que envíe notificaciones a Amazon SNS cuando se produzca un evento de escalado, como el lanzamiento de instancias o la terminación de instancias. Amazon SNS envía una notificación con información acerca de las instancias a la dirección de correo electrónico que ha especificado.

Para configurar las notificaciones de Amazon SNS para el grupo de Auto Scaling (consola)

1. Abra la consola Amazon EC2 en https://console.aws.amazon.com/ec2/y seleccione Auto Scaling Groups en el panel de navegación.
2. Seleccione la casilla situada junto al grupo de escalado automático.

   Se abre un panel dividido en la parte inferior de la página, que muestra información sobre el grupo seleccionado.
3. En la pestaña Activity (Actividad), seleccione Activity notifications (Notificaciones de actividad, Create notification (Crear notificación).
4. En el panel Create notifications, proceda del modo siguiente:

   a. En SNS Topic (Tema de SNS), seleccione el tema de SNS.

   b. En Event types (Tipos de eventos), seleccione los eventos para los que va enviar notificaciones.

   c. Seleccione Crear.

Para configurar las notificaciones de Amazon SNS para el grupo de Auto Scaling (AWS CLI)

Use el siguiente comando put-notification-configuration.

```
aws autoscaling put-notification-configuration --auto-scaling-group-name my-
asg --topic-arn arn --notification-types "autoscaling:EC2_INSTANCE_LAUNCH"
 "autoscaling:EC2_INSTANCE_TERMINATE"
```

Prueba de la notificación

Para generar una notificación para un evento de lanzamiento, actualice el grupo de Auto Scaling aumentando la capacidad deseada del grupo de Auto Scaling en 1. Usted recibe una notificación al cabo de unos minutos después de que se lance la instancia.

Para cambiar la capacidad deseada (consola)

1. Abra la consola Amazon EC2 en https://console.aws.amazon.com/ec2/y seleccione Auto Scaling Groups en el panel de navegación.
2. Seleccione la casilla situada junto al grupo de Auto Scaling.

   Se abre un panel dividido en la parte inferior de la página Auto Scaling groups (Grupos de Auto Scaling) que muestra información sobre el grupo seleccionado.
3. En la pestañaDetails (Detalles) elija Group details (Detalles de grupo), Edit (Editar).
4. En Desired capacity (Capacidad deseada), aumente el valor actual en 1. Si este valor supera el valor especificado en Maximum capacity (Capacidad máxima), también debe aumentar el valor de Maximum capacity (Capacidad máxima) en 1.
5. Elija Actualizar.
6. Después de unos minutos, recibirá una notificación del evento. Si no necesita la instancia adicional que lanzó para esta prueba, puede reducir el valor Desired capacity (Capacidad deseada) en 1. Después de unos minutos, recibirá una notificación del evento.

Eliminación de la configuración de notificaciones

Puede eliminar la configuración de notificaciones de Amazon EC2 Auto Scaling si ya no se utiliza.

Para eliminar la configuración de notificaciones de Amazon EC2 Auto Scaling (consola)

1. Abra la consola Amazon EC2 en https://console.aws.amazon.com/ec2/y seleccione Auto Scaling Groups en el panel de navegación.
2. Seleccione el grupo de escalado automático.
3. En la pestaña Actividad, seleccione la casilla situada junto a la notificación que desea eliminar y elija Acciones, Eliminar.

Para eliminar la configuración de notificaciones de Amazon EC2 Auto Scaling (AWS CLI)

Use el siguiente comando delete-notification-configuration.

```
aws autoscaling delete-notification-configuration --auto-scaling-group-name my-asg --
topic-arn arn
```

Para obtener información sobre cómo eliminar el tema de Amazon SNS y todas las suscripciones asociadas a un grupo de Auto Scaling, consulte Eliminación de una suscripción y un tema de Amazon SNS en la Guía para desarrolladores de Amazon Simple Notification Service.

Política de claves para un tema de Amazon SNS cifrado

El tema de Amazon SNS que especifique puede estar cifrado con una clave gestionada por el cliente creada con AWS Key Management Service. Para conceder permiso a Amazon EC2 Auto Scaling para publicar en temas cifrados, primero debe crear su clave de KMS y, a continuación, añadir la siguiente declaración a la política de la clave de KMS. Sustituya el ARN del ejemplo por el ARN del rol vinculado a servicios correspondiente que tiene permitido el acceso a la clave. Para obtener más información, consulte Configuración de permisos de AWS KMS en la Guía para desarrolladores de Amazon Simple Notification Service.

En este ejemplo, la declaración de política otorga AWSServiceRoleForAutoScalingpermisos al rol vinculado al servicio denominado para usar la clave administrada por el cliente. Para obtener más información sobre el rol vinculado a los servicios de Amazon EC2 Auto Scaling, consulte Roles vinculados a servicios de Amazon EC2 Auto Scaling.

```
{
  "Sid": "Allow service-linked role use of the customer managed key",
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::123456789012:role/aws-service-role/autoscaling.amazonaws.com/
AWSServiceRoleForAutoScaling"
  },
  "Action": [
    "kms:GenerateDataKey*",
    "kms:Decrypt"
  ],
  "Resource": "*"
}
```

Las claves de condición `aws:SourceArn` y `aws:SourceAccount` no se admiten en las políticas de claves que permiten a Amazon EC2 Auto Scaling publicar en temas cifrados.

AWS servicios integrados con Amazon EC2 Auto Scaling

Amazon EC2 Auto Scaling se puede integrar con AWS otros servicios. Consulte las siguientes opciones de integración para obtener más información sobre cómo funciona cada servicio con Amazon EC2 Auto Scaling.

Temas

- Reequilibrio de la capacidad en Auto Scaling para reemplazar las instancias de spot en riesgo
- Capacidad reservada en zonas de disponibilidad específicas con reserva de capacidad
- Cree grupos de Auto Scaling desde la línea de comandos usando AWS CloudShell
- Cree grupos de Auto Scaling con AWS CloudFormation
- Obtenga recomendaciones de tipos de instancias con AWS Compute Optimizer
- Utilice Elastic Load Balancing para distribuir el tráfico entrante de la aplicación en su grupo de escalado automático.
- Administración del flujo de tráfico con un grupo de destino de VPC Lattice
- Se usa EventBridge para gestionar eventos de Auto Scaling
- Proporcionar conectividad de red para sus instancias de Auto Scaling mediante Amazon VPC

Reequilibrio de la capacidad en Auto Scaling para reemplazar las instancias de spot en riesgo

El reequilibrio de capacidad en Auto Scaling permite mantener la disponibilidad de su carga de trabajo al sustituir de forma proactiva las instancias de spot que corren el riesgo de sufrir interrupciones.

Cuando las instancias de spot se encuentran en un riesgo elevado de interrupción, el servicio de spot de Amazon EC2 envía una recomendación de reequilibrio de instancias de EC2 a Amazon EC2 Auto Scaling. Si habilita el reequilibrio de la capacidad, Auto Scaling intentará reemplazar de forma proactiva las instancias de spot del grupo que han recibido una recomendación de reequilibrio de la instancia EC2. Esto le da la oportunidad de reequilibrar la carga de trabajo con nuevas instancias de spot que no tengan un riesgo elevado de interrupción.

Cuando no se usa el reequilibrio de la capacidad, Auto Scaling no reemplaza las instancias de spot hasta que el servicio de spot de Amazon EC2 interrumpe las instancias y se produzca un error en la

comprobación de estado. Antes de interrumpir una instancia, Amazon EC2 siempre proporciona una recomendación de reequilibrio de instancias de EC2 y un aviso de interrupción de las instancias de spot con dos minutos de antelación.

Contenido

- Descripción general de
- Comportamiento de reequilibrio de la capacidad
- Consideraciones
- Habilitar el reequilibrio de la capacidad para reemplazar de forma proactiva las instancias de spot en riesgo

Descripción general de

Para usar el reequilibrio de la capacidad con su grupo de escalado automático, los pasos básicos son:

1. Configure su grupo de escalado automático para utilizar varios tipos de instancia y zonas de disponibilidad. De esta forma, Amazon EC2 Auto Scaling puede examinar la capacidad disponible para instancias de spot en cada zona de disponibilidad. Para obtener más información, consulte Grupos de Auto Scaling con varios tipos de instancia y opciones de compra.
2. Agregue enlaces de ciclo de vida según sea necesario para cerrar correctamente la aplicación dentro de las instancias que reciben la notificación de reequilibrio. Para obtener más información, consulte Enlaces de ciclo de vida de Amazon EC2 Auto Scaling.

   A continuación, se indican algunos de los motivos por los que podría utilizar un enlace de ciclo de vida:

   - Apagar correctamente los trabajos de Amazon SQS
   - Completar la anulación del registro del sistema de nombres de dominio (DNS)
   - Extracción de registros del sistema o de aplicaciones y carga en Amazon Simple Storage Service (Amazon S3)
3. Desarrolle una acción personalizada para el enlace de ciclo de vida. Para invocar su acción personalizada lo antes posible, necesita saber cuándo una instancia está lista para ser terminada. Averígüelo detectando el estado del ciclo de vida de la instancia.
   - Para invocar una acción fuera de la instancia, escribe una EventBridge regla y automatiza la acción que se debe realizar cuando un patrón de eventos coincida con la regla.

- Para invocar una acción dentro de la instancia, configure la instancia para que ejecute un script de cierre y recupere el estado del ciclo de vida a través de los metadatos de la instancia.

Es fundamental diseñar la acción personalizada para que finalice en menos de dos minutos. Esto garantiza que haya tiempo suficiente para completar las tareas antes de la terminación de la instancia.

Una vez que complete estos pasos, podrá empezar a utilizar el reequilibrio de la capacidad.

Comportamiento de reequilibrio de la capacidad

Con el reequilibrio de la capacidad, Amazon EC2 Auto Scaling se comporta de la siguiente manera cuando una instancia recibe una recomendación de reequilibrio:

- Al lanzar la nueva instancia de spot, Amazon EC2 Auto Scaling espera hasta que la nueva instancia supera la comprobación de estado antes de efectuar la terminación de la instancia anterior. Al reemplazar más de una instancia, la terminación de cada instancia anterior comienza después de que la nueva instancia se lanzó y superó la comprobación de estado.
- Dado que Amazon EC2 Auto Scaling intenta lanzar nuevas instancias antes de terminar las anteriores, el hecho de satisfacer por completo o casi la capacidad máxima especificada podría impedir o detener completamente las actividades de reequilibrio. Para evitar este problema, Amazon EC2 Auto Scaling puede superar temporalmente el tamaño máximo del grupo hasta un 10 por ciento de la capacidad deseada.
- Si usted no agregó un enlace de ciclo de vida a su grupo de escalado automático, Amazon EC2 Auto Scaling comienza a terminar las instancias anteriores tan pronto como las nuevas instancias superen la comprobación de estado.
- Si agregó un enlace de ciclo de vida, esto prolonga el tiempo necesario para que empecemos a terminar las instancias anteriores en función del valor de tiempo de espera que especificó para el enlace de ciclo de vida.
- Si utiliza políticas de escalado o un escalado programado, las actividades de escalado se ejecutan en paralelo. Si una actividad de escalado está en curso y el grupo de escalado automático está por debajo de la nueva capacidad deseada, Amazon EC2 Auto Scaling se escala horizontalmente primero antes de terminar las instancias anteriores.

Si no hay capacidad para sus tipos de instancias en una zona de disponibilidad, Amazon EC2 Auto Scaling sigue intentando lanzar las instancias de spot en otras zonas de disponibilidad habilitadas hasta que logre efectuar la acción correctamente.

En el peor de los casos, si las nuevas instancias no se lanzan o si la comprobación de estado no se supera, Amazon EC2 Auto Scaling sigue intentando volver a lanzarlas. Mientras está tratando de lanzar nuevas instancias, las anteriores finalmente se verán interrumpidas y se terminarán a la fuerza previo aviso de interrupción tras dos minutos.

Consideraciones

Tenga en cuenta lo siguiente cuando use el reequilibrio de la capacidad:

Diseño de su aplicación para que sea tolerante a las interrupciones de spot

La aplicación debería poder gestionar cambios dinámicos en el número de instancias y la posibilidad de que una instancia de spot se interrumpa antes. Por ejemplo, si su grupo de escalado automático está detrás de un equilibrador de carga de Elastic Load Balancing, Amazon EC2 Auto Scaling espera a que el registro de la instancia se anule en el equilibrador de carga antes de llamar a su enlace de ciclo de vida. Si el tiempo para anular el registro de la instancia y completar la acción de ciclo de vida tarda demasiado, la instancia puede interrumpirse mientras Amazon EC2 Auto Scaling espera a que su acción de ciclo de vida se complete antes de terminar la instancia.

Amazon EC2 no siempre puede enviar la señal de recomendación de reequilibrio antes del aviso de interrupción de dos minutos de instancia de spot. En ocasiones, la señal de recomendación de reequilibrio llega al mismo tiempo que el aviso de interrupción de dos minutos. Cuando esto ocurre, Amazon EC2 Auto Scaling llama al enlace de ciclo de vida e intenta lanzar una nueva instancia de spot de inmediato.

Evite un riesgo elevado de interrupción de instancias de spot de reemplazo

Las instancias de spot de reemplazo podrían correr un riesgo elevado de interrupción si utiliza la estrategia de asignación `lowest-price`. Esto se debe a que siempre lanzamos instancias en el grupo de menor precio que tiene capacidad disponible en ese momento, incluso si es probable que las instancias de spot de reemplazo se interrumpan poco después de lanzarse. Para evitar un alto riesgo de interrupción, es recomendable no utilizar la estrategia de asignación del `lowest-price`. En su lugar, recomendamos la estrategia de asignación del `price-capacity-optimized`. Esta estrategia lanza instancias de spot de reemplazo en grupos de

spot que tienen menos probabilidades de interrupción y el precio más bajo posible. Por lo tanto, es menos probable que se interrumpan en un futuro cercano.

Amazon EC2 Auto Scaling solo lanzará una nueva instancia si la disponibilidad es igual o superior

Uno de los objetivos del reequilibrio de la capacidad es mejorar la disponibilidad de una instancia de spot. Si una instancia de spot existente recibe una recomendación de reequilibrio, Amazon EC2 Auto Scaling solo lanzará una nueva instancia si la nueva instancia ofrece la misma o mejor disponibilidad que la existente. Si el riesgo de interrupción de una nueva instancia es peor que el de la instancia existente, Amazon EC2 Auto Scaling no lanzará ninguna instancia nueva. Sin embargo, Amazon EC2 Auto Scaling seguirá evaluando los grupos de capacidades de spot con base en la información proporcionada por el servicio de spot de Amazon EC2 y lanzará una nueva instancia si mejora la disponibilidad.

Existe la posibilidad de que la instancia existente se interrumpa sin que Amazon EC2 Auto Scaling lance una nueva instancia de forma proactiva. Cuando esto suceda, Amazon EC2 Auto Scaling intenta lanzar una nueva instancia tan pronto como reciba el aviso de interrupción de la instancia de spot. Esto ocurre independientemente de si la nueva instancia tiene un alto riesgo de interrupción.

El reequilibrio de la capacidad no aumenta la tasa de interrupciones de las instancias de spot

Cuando habilita el reequilibrio de capacidad, no aumenta la tasa de interrupciones de las instancias de spot (el número de instancias de spot que se reclaman cuando Amazon EC2 necesita recuperar la capacidad). Sin embargo, si el reequilibrio de capacidad detecta que una instancia está en riesgo de interrupción, Amazon EC2 Auto Scaling intentará lanzar inmediatamente una nueva instancia. Por lo tanto, se podrían reemplazar más instancias en lugar de esperar a que Amazon EC2 Auto Scaling lance una nueva después de que se interrumpa la instancia en riesgo.

Si bien es posible que se reemplacen más instancias con el reequilibrio de la capacidad habilitado, se beneficia de ser proactivo en lugar de reactivo. Esto le da más tiempo para tomar medidas antes de que sus instancias se interrumpan. Con un aviso de interrupción de instancias de spot, normalmente solo dispone de dos minutos para apagar correctamente la instancia. Dado que el reequilibrio de la capacidad lanza una nueva instancia por adelantado, le da a los procesos existentes una mejor oportunidad de completarse en la instancia en riesgo. También puede iniciar los procedimientos de apagado de la instancia, evitar que se programen nuevos trabajos en la instancia en riesgo y preparar la instancia recién lanzada para que se haga cargo de la aplicación. Con el reemplazo proactivo del reequilibrio de la capacidad, usted se beneficia de una continuidad estable.

El siguiente ejemplo teórico demuestra los riesgos y beneficios del uso del reequilibrio de la capacidad:

- 14:00: se recibe una recomendación de reequilibrio para la instancia A. Amazon EC2 Auto Scaling inmediatamente intenta lanzar una instancia B de reemplazo, lo que le da tiempo para iniciar los procedimientos de apagado.
- 14:30: se recibe una recomendación de reequilibrio para la instancia B, que es reemplazada por la instancia C. Esto le da tiempo para iniciar los procedimientos de apagado.
- 14:32: si el reequilibrio de la capacidad no estuviera habilitado y si se hubiera recibido un aviso de interrupción de la instancia de spot a las 14:32 para la instancia A, usted solo habría tenido dos minutos para actuar. Sin embargo, la instancia A habría seguido ejecutándose hasta ese momento.

Habilitar el reequilibrio de la capacidad para reemplazar de forma proactiva las instancias de spot en riesgo

Puede usar Consola de administración de AWS o AWS CLI para habilitar el reequilibrio de capacidad para su grupo de Auto Scaling. Cuando se habilita el reequilibrio de la capacidad, Amazon EC2 Auto Scaling intentará reemplazar de forma proactiva las instancias de spot en su grupo que han recibido una recomendación de reequilibrio de instancias de EC2.

Habilitar el reequilibrio de la capacidad (consola)

Puede habilitar o desactivar el reequilibrio de la capacidad al crear o actualizar un grupo de escalado automático.

Para habilitar el reequilibrio de la capacidad para un nuevo grupo de escalado automático

1. Abra la consola Amazon EC2 en https://console.aws.amazon.com/ec2/y seleccione Auto Scaling Groups en el panel de navegación.
2. Elija Create Auto Scaling group (Crear grupo de escalado automático).
3. Para el paso 1: Elegir la plantilla de lanzamiento o la configuración, ingrese un nombre para el grupo de escalado automático, elija una plantilla de lanzamiento y, a continuación, elija Siguiente para continuar con el próximo paso.
4. Para el paso 2: Elegir las opciones de lanzamiento de instancias, en Requisitos de tipo de instancias, elija la configuración para crear un grupo de instancias mixtas. Esto incluye los tipos de instancias que puede lanzar, las opciones de compra de instancias y las estrategias

de asignación para las instancias de spot y bajo demanda. De forma predeterminada, estas opciones no están configuradas. Para configurarlas, debe seleccionar Override launch template (Anular plantilla de lanzamiento). Para obtener más información sobre cómo crear un grupo de instancias mixtas, consulte Grupos de Auto Scaling con varios tipos de instancia y opciones de compra.

5. En Red, elija las opciones que desee. Compruebe que las subredes que desea utilizar se encuentran en diferentes zonas de disponibilidad.
6. En la sección Estrategias de asignación, elija una estrategia de asignación de spot. Para habilitar o deshabilitar el reequilibrio de la capacidad, seleccione o desmarque la casilla debajo de Reequilibrio de la capacidad. Esta opción solo aparece cuando uno solicita un porcentaje del grupo de escalado automático que se lanzará como instancias de spot en la sección Opciones de compra de instancias.
7. Cree el grupo de escalado automático.
8. (Opcional) Añada enlaces de ciclo de vida según sea necesario. Para obtener más información, consulte Incorporación de los enlaces de ciclo de vida en su grupo de escalado automático.

Habilitación o deshabilitación del reequilibrio de la capacidad para un grupo de escalado automático existente

1. Abra la consola Amazon EC2 en https://console.aws.amazon.com/ec2/y seleccione Auto Scaling Groups en el panel de navegación.
2. Seleccione la casilla situada junto al grupo de escalado automático. Se abre un panel dividido en la parte inferior de la página.
3. En la pestaña Details (Detalles), elija Allocation strategies (Estrategias de asignación) y Edit (Editar).
4. En la sección Estrategias de asignación, habilite o deshabilite el reequilibrio de la capacidad seleccionando o desmarcando la casilla debajo de Reequilibrio de la capacidad.
5. Elija Actualizar.

Habilitar el reequilibrio de la capacidad (AWS CLI)

Los siguientes ejemplos muestran cómo utilizar el AWS CLI para activar y desactivar el reequilibrio de capacidad.

Utilice el update-auto-scaling-groupcomando create-auto-scaling-groupo con el siguiente parámetro:

- `--capacity-rebalance` / `--no-capacity-rebalance`: valor booleano que indica si el reequilibrio de la capacidad está habilitado.

Antes de ejecutar el create-auto-scaling-groupcomando, necesitará el nombre de una plantilla de lanzamiento que esté configurada para usarse con un grupo de Auto Scaling. Para obtener más información, consulte Creación de una plantilla de lanzamiento para un grupo de Auto Scaling.

> **Note**
>
> En los procedimientos siguientes se muestra cómo utilizar un archivo de configuración con formato JSON o YAML. Si utiliza la AWS CLI versión 1, debe especificar un archivo de configuración con formato JSON. Si usa la AWS CLI versión 2, puede especificar un archivo de configuración formateado en YAML o JSON.

JSON

Para crear y configurar un nuevo grupo de escalado automático

- Use el siguiente create-auto-scaling-groupcomando para crear un nuevo grupo de Auto Scaling y habilitar el reequilibrio de capacidad. Este comando hace referencia a un archivo JSON como único parámetro de su grupo de escalado automático.

  ```
  aws autoscaling create-auto-scaling-group --cli-input-json file://~/config.json
  ```

  Si aún no tiene un archivo de configuración de la CLI que especifique una política de instancias mixtas, cree uno.

  Agregue la siguiente línea al objeto JSON de nivel superior en el archivo de configuración.

  ```
  {
      "CapacityRebalance": true
  }
  ```

  A continuación se muestra un ejemplo de un archivo `config.json`.

  ```
  {
      "AutoScalingGroupName": "my-asg",
      "DesiredCapacity": 12,
  ```

```
    "MinSize": 12,
    "MaxSize": 15,
    "CapacityRebalance": true,
    "MixedInstancesPolicy": {
        "InstancesDistribution": {
            "OnDemandBaseCapacity": 0,
            "OnDemandPercentageAboveBaseCapacity": 25,
            "SpotAllocationStrategy": "price-capacity-optimized"
        },
        "LaunchTemplate": {
            "LaunchTemplateSpecification": {
                "LaunchTemplateName": "my-launch-template",
                "Version": "$Default"
            },
            "Overrides": [
                {
                    "InstanceType": "c5.large"
                },
                {
                    "InstanceType": "c5a.large"
                },
                {
                    "InstanceType": "m5.large"
                },
                {
                    "InstanceType": "m5a.large"
                },
                {
                    "InstanceType": "c4.large"
                },
                {
                    "InstanceType": "m4.large"
                },
                {
                    "InstanceType": "c3.large"
                },
                {
                    "InstanceType": "m3.large"
                }
            ]
        }
    },
    "TargetGroupARNs": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:targetgroup/my-alb-target-group/943f017f100becff",
```

```
    "VPCZoneIdentifier": "subnet-5ea0c127,subnet-6194ea3b,subnet-c934b782"
}
```

YAML

Para crear y configurar un nuevo grupo de escalado automático

- Use el siguiente create-auto-scaling-groupcomando para crear un nuevo grupo de Auto Scaling y habilitar el reequilibrio de capacidad. Este comando hace referencia a un archivo YAML como único parámetro de su grupo de escalado automático.

  ```
  aws autoscaling create-auto-scaling-group --cli-input-yaml file://~/config.yaml
  ```

  Agregue la siguiente línea al archivo de configuración con formato YAML.

  ```
  CapacityRebalance: true
  ```

  A continuación se muestra un ejemplo de un archivo `config.yaml`.

  ```
  ---
  AutoScalingGroupName: my-asg
  DesiredCapacity: 12
  MinSize: 12
  MaxSize: 15
  CapacityRebalance: true
  MixedInstancesPolicy:
    InstancesDistribution:
      OnDemandBaseCapacity: 0
      OnDemandPercentageAboveBaseCapacity: 25
      SpotAllocationStrategy: price-capacity-optimized
    LaunchTemplate:
      LaunchTemplateSpecification:
        LaunchTemplateName: my-launch-template
        Version: $Default
      Overrides:
      - InstanceType: c5.large
      - InstanceType: c5a.large
      - InstanceType: m5.large
      - InstanceType: m5a.large
      - InstanceType: c4.large
  ```

```
    - InstanceType: m4.large
    - InstanceType: c3.large
    - InstanceType: m3.large
TargetGroupARNs:
- arn:aws:elasticloadbalancing:us-west-2:123456789012:targetgroup/my-alb-target-group/943f017f100becff
VPCZoneIdentifier: subnet-5ea0c127,subnet-6194ea3b,subnet-c934b782
```

Para habilitar el reequilibrio de la capacidad para un grupo de escalado automático existente

- Use el siguiente update-auto-scaling-groupcomando para habilitar el reequilibrio de capacidad.

```
aws autoscaling update-auto-scaling-group --auto-scaling-group-name my-asg \
  --capacity-rebalance
```

Para verificar que el reequilibrio de la capacidad esté habilitado para un grupo de escalado automático

- Utilice el siguiente describe-auto-scaling-groupscomando para comprobar que el reequilibrio de capacidad esté activado y para ver los detalles.

```
aws autoscaling describe-auto-scaling-groups --auto-scaling-group-name my-asg
```

A continuación, se muestra un ejemplo de respuesta.

```
{
    "AutoScalingGroups": [
        {
            "AutoScalingGroupName": "my-asg",
            "AutoScalingGroupARN": "arn",
            ...
            "CapacityRebalance": true
        }
    ]
}
```

Para desactivar el reequilibrio de la capacidad

Utilice el update-auto-scaling-groupcomando con la `--no-capacity-rebalance` opción de deshabilitar el reequilibrio de capacidad.

```
aws autoscaling update-auto-scaling-group --auto-scaling-group-name my-asg \
  --no-capacity-rebalance
```

Recursos relacionados

Para obtener más información sobre el reequilibrio de capacidad, consulte Gestionar proactivamente el ciclo de vida de las instancias puntuales mediante la nueva función de reequilibrio de capacidad para Amazon EC2 Auto Scaling en el blog de informática. AWS

Para obtener más información acerca de las recomendaciones de reequilibrio de instancias de EC2, consulte Recomendación de reequilibrio de instancias de EC2 en la Guía del usuario de Amazon EC2.

Para obtener más información acerca de los enlaces de ciclo de vida, consulte los siguientes recursos.

- Tutorial: Configuración de un enlace de ciclo de vida que invoca una función Lambda(usando) EventBridge
- Tutorial: Uso del script de datos y los metadatos de la instancia para recuperar el estado del ciclo de vida

Limitaciones

- Amazon EC2 Auto Scaling puede reemplazar la instancia que recibe la notificación de reequilibrio solo si la instancia no está protegida contra la reducción horizontal. Sin embargo, la protección de reducir horizontalmente no impide la terminación debido a una interrupción de spot. Para obtener más información, consulte Uso de la protección frente a la reducción horizontal de instancias para controlar la terminación de instancias.
- La compatibilidad con el reequilibrio de la capacidad está disponible en todas las Regiones de AWS comerciales donde está disponible Amazon EC2 Auto Scaling, excepto la región de Oriente Medio (EAU).

Capacidad reservada en zonas de disponibilidad específicas con reserva de capacidad

Las reservas de capacidad bajo demanda de Amazon EC2 permiten reservar capacidad de cómputo para zonas de disponibilidad específicas. Para comenzar a utilizar reservas de capacidad con Auto Scaling, primero cree una reserva de capacidad o un grupo de reserva de capacidad en una zona de disponibilidad específica. Luego, puede agregar una preferencia de reserva de capacidad a su grupo de escalado automático al crearlo o al actualizar un grupo existente.

Para crear una reserva de capacidad, consulte Crear una reserva de capacidad en la Guía del usuario de Amazon EC2. Y para crear un grupo de reserva de capacidad, consulte Crear un grupo de reservas de capacidad en la Guía del usuario de Amazon EC2.

Preferencia de reserva de capacidad

La preferencia de reserva de capacidad permite usar las reservas de capacidad de manera eficiente al priorizar la capacidad reservada en una reserva de capacidad antes de usar la capacidad bajo demanda. Puede seleccionar una de las siguientes opciones de preferencia de reserva de capacidad:

- Predeterminado: Auto Scaling utiliza la preferencia de reserva de capacidad de su plantilla de lanzamiento o una reserva de capacidad abierta.
- Ninguno: Auto Scaling no lanzará instancias en una reserva de capacidad. Las instancias solo se ejecutarán en capacidad bajo demanda.
- Solo reservas de capacidad: Auto Scaling solo lanzará instancias en una reserva de capacidad o un grupo de reserva de capacidad. No se lanzarán las instancias si no hay capacidad disponible.
- Reservas de capacidad primero: Auto Scaling lanzará instancias en una reserva de capacidad o un grupo de reserva de capacidad. Las instancias se ejecutarán en capacidad bajo demanda si no hay capacidad disponible.

Si selecciona solo reservas de capacidad o reservas de capacidad primero, puede especificar un objetivo de reserva de capacidad.

> **Note**
>
> Es obligatorio seleccionar una preferencia de reserva de capacidad. El objetivo de reserva de capacidad es opcional.

Aspectos a tener en cuenta sobre las preferencias de reserva de capacidad y las plantillas de lanzamiento

Es importante recordar lo siguiente si selecciona solo reservas de capacidad o reservas de capacidad primero:

- Si selecciona solo reservas de capacidad o reservas de capacidad primero, Auto Scaling utilizará el objetivo de reserva de capacidad especificado en el grupo de escalado automático en lugar del objetivo de reserva de capacidad de la plantilla de lanzamiento.
- Si selecciona solo reservas de capacidad o reservas de capacidad primero y no especifica un objetivo de reserva de capacidad, Auto Scaling utilizará el objetivo de reserva de capacidad de la plantilla de lanzamiento o una reserva de capacidad abierta.

Especificación del objetivo de reserva de capacidad

Si selecciona solo reservas de capacidad o reservas de capacidad primero, estarán disponibles las siguientes opciones del objetivo de reserva de capacidad:

- Abierta: Auto Scaling lanzará instancias en cualquier reserva de capacidad abierta. Si seleccionó solo reservas de capacidad y la capacidad no está disponible, las instancias no se lanzarán. Pero si seleccionó reservas de capacidad primero y la capacidad no está disponible, las instancias se lanzarán en la capacidad bajo demanda.
- Especificar reserva de capacidad: Auto Scaling lanzará las instancias en una reserva de capacidad especificada. Si seleccionó solo reservas de capacidad y la capacidad no está disponible, las instancias no se lanzarán. Pero si seleccionó reservas de capacidad primero y la capacidad no está disponible, las instancias se lanzarán en la capacidad bajo demanda.
- Especificar el grupo de recursos de reserva de capacidad: Auto Scaling lanzará las instancias en una reserva de capacidad abierta en un grupo de recursos de reserva de capacidad especificado. Si seleccionó solo reservas de capacidad y la capacidad no está disponible, las instancias no se lanzarán. Pero si seleccionó reservas de capacidad primero y la capacidad no está disponible, las instancias se lanzarán en la capacidad bajo demanda.

Equilibrio de la zona de disponibilidad y las reservas de capacidad

Auto Scaling le da prioridad al equilibrio de la zona de disponibilidad incluso cuando se utilizan las preferencias de reservas de capacidad. Esto significa:

- Con`capacity-reservations-first`: el grupo Auto Scaling distribuirá AZs primero las instancias de manera uniforme y, a continuación, utilizará las reservas de capacidad cuando estén disponibles. Si las reservas de capacidad no están disponibles en una zona de disponibilidad, las instancias se lanzarán bajo demanda para mantener el equilibrio de la zona de disponibilidad.
- Con `capacity-reservations-only`: el grupo de escalado automático solo usará reservas de capacidad. Esto podría provocar una distribución desigual de las instancias en AZs función de dónde estén disponibles las reservas de capacidad.

Ejemplo

Si tiene 10 reservas de capacidad en AZ-a, 3 en AZ-b, 1 en AZ-c y una capacidad deseada de 9 instancias:

- Con `capacity-reservations-first` para 9 instancias, se obtendrán 3 instancias por zona de disponibilidad (manteniendo el equilibrio de la zona de disponibilidad), y algunas instancias podrían ejecutarse bajo demanda.
- Con `capacity-reservations-only` para 9 instancias provocará una distribución desigual según las reservas de capacidad disponibles.

Utilizar la preferencia de reserva de capacidad con el grupo de escalado automático

Para usar las reservas de capacidad con su grupo de escalado automático, primero debe crear una reserva de capacidad o un grupo de recursos de reserva de capacidad. Luego, puede agregar una preferencia de reserva de capacidad a su grupo de escalado automático al crearlo o al actualizar un grupo existente.

Para crear una reserva de capacidad, consulte Crear una reserva de capacidad en la Guía del usuario de Amazon EC2. Y para crear un grupo de reserva de capacidad, consulte Crear un grupo de reservas de capacidad en la Guía del usuario de Amazon EC2.

Utilice uno de los siguientes métodos para usar la preferencia de reserva de capacidad al crear o editar un grupo de escalado automático.

Console

Cómo usar la preferencia de reserva de capacidad en un grupo nuevo (consola)

1. Siga las instrucciones Creación de un grupo de Auto Scaling mediante el asistente de lanzamiento de Amazon EC2 y complete cada paso del procedimiento, hasta el paso 3.
2. En la página Configurar el tamaño y la escala del grupo, en Ajustes de capacidad adicionales, Preferencia de reserva de capacidad, seleccione una preferencia de reserva de capacidad. Para obtener información sobre la preferencia de reserva de capacidad, consulte Preferencia de reserva de capacidad.
3. Continúe con los pasos en Creación de un grupo de Auto Scaling mediante el asistente de lanzamiento de Amazon EC2.

AWS CLI

Para usar la preferencia de reserva de capacidad en un grupo nuevo (AWS CLI)

Agregue el parámetro `--capacity-reservation-specification` al comando create-auto-scaling-group.

1. Especifique la preferencia de reserva de capacidad. Para obtener más información, consulte Preferencia de reserva de capacidad.
2. Especifique un objetivo de reserva de capacidad. Si selecciona solo reservas de capacidad o reservas de capacidad primero y no especifica un objetivo de reserva de capacidad, Auto Scaling utilizará el objetivo de reserva de capacidad de la plantilla de lanzamiento o una reserva de capacidad abierta.

Console

Cómo usar la preferencia de reserva de capacidad en un grupo existente (consola)

1. Abra la consola Amazon EC2 en https://console.aws.amazon.com/ec2/y seleccione Auto Scaling Groups en el panel de navegación.
2. En la barra de navegación de la parte superior de la pantalla, elija la misma Región de AWS que utilizó cuando creó el grupo de escalado automático.

3. Seleccione la casilla situada junto al grupo de escalado automático.

   Se abre un panel dividido en la parte inferior de la página.
4. En la pestaña Detalles, en la sección Preferencia de reserva de capacidad, seleccione Editar.
5. En Ajustes de capacidad adicional, Preferencia de reserva de capacidad, seleccione una preferencia de reserva de capacidad. Para obtener información sobre la preferencia de reserva de capacidad, consulte Preferencia de reserva de capacidad.
6. Elija Actualizar.

AWS CLI

Cómo usar la preferencia de reserva de capacidad en un grupo existente (AWS CLI)

Agregue el parámetro `--capacity-reservation-specification` al comando update-auto-scaling-group.

1. Especifique la preferencia de reserva de capacidad. Para obtener más información, consulte Preferencia de reserva de capacidad.
2. Especifique un objetivo de reserva de capacidad. Si selecciona solo reservas de capacidad o reservas de capacidad primero y no especifica un objetivo de reserva de capacidad, Auto Scaling utilizará el objetivo de reserva de capacidad de la plantilla de lanzamiento o una reserva de capacidad abierta.

Reservas de capacidad interrumpible con Auto Scaling de EC2

Las reservas de capacidad interrumpible representan la capacidad sobrante que le prestan los propietarios de reservas de capacidad bajo demanda de su organización. AWS Estas reservas son adecuadas para cargas de trabajo interrumpibles, ya que el propietario de la capacidad puede recuperar la capacidad en cualquier momento con un aviso de interrupción de 2 minutos, tras lo cual se dan por terminadas las instancias. Para obtener información general sobre las reservas de capacidad interrumpible y su funcionamiento, consulte las reservas de capacidad interrumpible en la Guía del usuario de Amazon EC2.

Para empezar, cree una reserva interrumpible asignando la capacidad no utilizada de cualquier reserva de capacidad bajo demanda de su cuenta. A continuación, debe crear la plantilla de lanzamiento, especificar el identificador de reserva de capacidad interrumpible y el tipo de mercado

correspondiente. Por último, cree su grupo de Auto Scaling para usar la plantilla de lanzamiento que ha creado.

> **Note**
>
> - No se admiten grupos de instancias mixtas con reservas de capacidad interrumpibles.
> - Las reservas de capacidad interrumpibles aparecen en su cuenta con un nuevo atributo `interruptible` establecido en. `true`

Temas

- Directrices operativas
- Especifica una reserva de capacidad interrumpible en tu plantilla de lanzamiento
- Limitaciones
- Recursos relacionados

Directrices operativas

Las siguientes son pautas operativas básicas que debe seguir cuando utilice reservas de capacidad interrumpible con un grupo de Auto Scaling.

- Las instancias lanzadas a partir de reservas de capacidad interrumpible se pueden cancelar con un aviso de 2 minutos cuando se recupere la capacidad. Diseñe sus aplicaciones de manera que se puedan detener sin problemas.
- Amazon EC2 envía EventBridge las notificaciones 2 minutos antes de la finalización de las instancias. Si bien Auto Scaling no reacciona automáticamente ante estos eventos, puede configurar una automatización personalizada para responder a los avisos de interrupción.
- Cuando las instancias se cancelan debido a la recuperación de capacidad, el comprobador de estado de Auto Scaling detectará la terminación y lanzará automáticamente las instancias de reemplazo según la configuración del grupo.
- Seleccione una versión específica de la plantilla de lanzamiento en lugar de la `$Latest` versión `$Default` o para garantizar la coherencia.

> **Note**
>
> Cuando se recupera una instancia de reserva de capacidad interrumpible, las actividades de escalado de su grupo de Auto Scaling indicarán:. `taken out of service in response to an Amazon EC2 health check that indicated it had been terminated or stopped` Auto Scaling intentará reemplazar la instancia de la misma manera que lo hace con cualquier instancia que no pase una comprobación de estado.

Especifica una reserva de capacidad interrumpible en tu plantilla de lanzamiento

Para crear una plantilla de lanzamiento destinada a una reserva de capacidad interrumpible específica para su grupo de Auto Scaling, utilice uno de los siguientes métodos:

Console

Para especificar una reserva de capacidad interrumpible en la plantilla de lanzamiento (consola)

1. Abra la consola de Amazon EC2 en https://console.aws.amazon.com/ec2/.
2. En la barra de navegación superior, selecciona el Región de AWS lugar donde tienes tu reserva de capacidad interrumpible.
3. En el panel de navegación, en Instances, seleccione Launch Templates.
4. Seleccione Crear plantilla de lanzamiento y cree la plantilla de lanzamiento. Incluya el ID de la AMI, el tipo de instancia y cualquier otra configuración de la plantilla de lanzamiento según sea necesario.
5. Expanda la sección Detalles avanzados para ver la configuración avanzada.
6. Como opción de compra, elija Reserva de capacidad interrumpible.
7. Para la reserva de capacidad, elija Target por ID y, a continuación, en Reserva de capacidad (Target by ID), elija la ID de reserva de capacidad de una reserva de capacidad interrumpible existente.
8. Cuando haya terminado, seleccione Crear plantilla de lanzamiento.

Para obtener ayuda para crear un grupo de escalado automático con una plantilla de lanzamiento, consulte Creación de un grupo de Auto Scaling mediante una plantilla de lanzamiento.

AWS CLI

Para especificar una reserva de capacidad interrumpible en la plantilla de lanzamiento ()AWS CLI

Utilice el siguiente create-launch-templatecomando para crear una plantilla de lanzamiento que especifique un ID de reserva de capacidad interrumpible existente. Reemplace cada marcador de posición de entrada del usuario con información propia.

```
aws ec2 create-launch-template --launch-template-name my-template-for-interruptible-cr \
  --version-description AutoScalingVersion1 --region us-east-2 \
  --launch-template-data file://config.json
```

> **Tip**
>
> Si este comando arroja un error, asegúrese de haber actualizado la versión AWS CLI local a la última versión.

Contenido de `config.json`:

```
{
    "ImageId": "ami-04d5cc9b88example",
    "InstanceType": "t3.nano",
    "SecurityGroupIds": [
        "sg-903004f88example"
    ],
    "KeyName": "MyKeyPair",
    "InstanceMarketOptions": {
        "MarketType": "interruptible-capacity-reservation"
    },
    "CapacityReservationSpecification": {
        "CapacityReservationTarget": {
            "CapacityReservationId": "cr-02168da1478b509e0"
        }
    }
}
```

A continuación, se muestra un ejemplo de la salida:

```
{
    "LaunchTemplate": {
```

```
        "LaunchTemplateId": "lt-068f72b724example",
        "LaunchTemplateName": "my-template-for-interruptible-cr",
        "CreateTime": "2023-10-27T15:12:44.000Z",
        "CreatedBy": "arn:aws:iam::123456789012:user/Bob",
        "DefaultVersionNumber": 1,
        "LatestVersionNumber": 1
    }
}
```

Puede usar el siguiente describe-launch-template-versionscomando para verificar el identificador de reserva de capacidad interrumpible asociado a la plantilla de lanzamiento:

```
aws ec2 describe-launch-template-versions --launch-template-names my-template-for-interruptible-cr \
  --region us-east-2
```

Configuración a nivel de grupo de Auto Scaling

Si lo desea, puede especificar la configuración de reserva de capacidad en el nivel de grupo de Auto Scaling para anular la configuración de la plantilla de lanzamiento:

- Preferencia de reserva de capacidad: solo `capacity-reservations-only` se admite para reservas de capacidad interrumpibles. No se admite esta `capacity-reservations-first` preferencia.
- Especificación objetivo: puede especificar un identificador de reserva de capacidad interrumpible diferente a nivel de grupo para anular la configuración de la plantilla de lanzamiento.

Limitaciones

- Las políticas de instancias mixtas no son compatibles con las reservas de capacidad interrumpible.
- Debe especificar el tipo de mercado correcto (`interruptible-capacity-reservation`) en la plantilla de lanzamiento junto con el identificador de reserva de capacidad.

Recursos relacionados

- Para obtener información sobre la creación y administración de reservas de capacidad interrumpible, consulte Reservas de capacidad interrumpible en la Guía del usuario de Amazon EC2.

- Para obtener información general sobre el uso de reservas de capacidad con Auto Scaling, consulteCapacidad reservada en zonas de disponibilidad específicas con reserva de capacidad.
- Para obtener información sobre las comprobaciones de estado de los grupos de Auto Scaling, consulte Comprobaciones de estado para instancias de un grupo de Auto Scaling.

Cree grupos de Auto Scaling desde la línea de comandos usando AWS CloudShell

Si es compatible Regiones de AWS, puede ejecutar AWS CLI comandos utilizando AWS CloudShell un shell preautenticado y basado en un navegador que se inicia directamente desde el. Consola de administración de AWS Puedes ejecutar AWS CLI comandos en los servicios mediante el shell que prefieras (shell Bash o Z). PowerShell

Puedes AWS CloudShell lanzarlos desde uno de los dos métodos siguientes: Consola de administración de AWS

- Seleccione el AWS CloudShell icono de la barra de navegación de la consola. Se encuentra a la derecha del cuadro de búsqueda.
- Utilice el cuadro de búsqueda de la barra de navegación de la consola para buscar la CloudShellopción CloudShelly, a continuación, seleccionarla.

Cuando se AWS CloudShell abre por primera vez en una nueva ventana del navegador, aparece un panel de bienvenida con una lista de las funciones principales. Después de cerrar este panel, se proporcionan actualizaciones de estado mientras el shell configura y reenvía las credenciales de la consola. Cuando aparece el símbolo del sistema, el shell está listo para la interacción.

Para obtener más información acerca este servicio, consulte la Guía del usuario de AWS CloudShell.

Cree grupos de Auto Scaling con AWS CloudFormation

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

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

Amazon EC2 Auto CloudFormation Scaling y plantillas

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

Para empezar a crear sus propias plantillas de pila para Amazon EC2 Auto Scaling, haga las siguientes tareas:

- Cree una plantilla de lanzamiento utilizando AWS::EC2::LaunchTemplate.
- Cree un grupo de Auto Scaling mediante AWS::AutoScaling::AutoScalingGroup AWS::AutoScaling::AutoScaling .

Para ver un tutorial que muestra cómo implementar un grupo de escalado automático detrás de un Equilibrador de carga de aplicación, consulte Tutorial: Cree un servidor web escalable con balanceadores de carga en la Guía del usuario de AWS CloudFormation .

Puede encontrar ejemplos útiles adicionales de fragmentos de plantillas con las que se crean grupos de escalado automático y recursos relacionados en las siguientes secciones de la Guía del usuario de AWS CloudFormation .

- Referencia de tipo de recurso de Amazon EC2 Auto Scaling
- Configure los recursos de Amazon EC2 Auto Scaling con AWS CloudFormation

Obtenga más información sobre CloudFormation

Para obtener más información CloudFormation, consulte los siguientes recursos:

- AWS CloudFormation
- AWS CloudFormation Guía del usuario
- CloudFormation Referencia de la API

- AWS CloudFormation Guía del usuario de la interfaz de línea de comandos

Obtenga recomendaciones de tipos de instancias con AWS Compute Optimizer

AWS proporciona recomendaciones de tipos de instancias de Amazon EC2 para ayudarlo a mejorar el rendimiento, ahorrar dinero o ambas cosas, mediante el uso de funciones impulsadas por. AWS Compute Optimizer Puede utilizar estas recomendaciones para decidir si desea pasar a un nuevo tipo de instancia en su grupo de escalado automático.

Para hacer recomendaciones, Compute Optimizer analiza las especificaciones de instancia existentes y el historial de métricas recientes. A continuación, los datos compilados se utilizan para recomendar qué tipos de instancia de Amazon EC2 se optimizan mejor para gestionar la carga de trabajo de rendimiento existente. Las recomendaciones se devuelven junto con el precio de la instancia por hora.

> **Note**
>
> Para obtener recomendaciones de Compute Optimizer, primero debe darse de alta en Compute Optimizer. Para obtener más información, consulte el Tutorial de introducción a AWS Compute Optimizer en la Guía del usuario de AWS Compute Optimizer .

Contenido

- Limitaciones
- Hallazgos
- Ver recomendaciones
- Consideraciones para evaluar las recomendaciones

Limitaciones

Compute Optimizer genera recomendaciones para instancias en grupos de Auto Scaling que están configurados para lanzar y ejecutar tipos de instancias M, C, R, T y X. Sin embargo, no genera recomendaciones para los tipos de instancias -g que utilizan procesadores AWS Graviton2 (por ejemplo, C6g) ni para los tipos de instancias -n que tienen un rendimiento de ancho de banda de red superior (por ejemplo, M5n).

Los grupos Auto Scaling también deben configurarse para ejecutar un único tipo de instancia (es decir, no hay tipos de instancia mixtos), no deben tener una política de escalado asociada a ellos y tener los mismos valores para la capacidad deseada, mínima y máxima (es decir, un grupo de escalado automático con un número fijo de instancias). Compute Optimizer genera recomendaciones para instancias en grupos de Auto Scaling que cumplen todo de estos requisitos de configuración.

Hallazgos

Compute Optimizer clasifica sus hallazgos para grupos de Auto Scaling de la siguiente manera:

- No optimizado: se considera que un grupo de escalado automático no está optimizado cuando Compute Optimizer ha identificado una recomendación que puede proporcionar un mejor rendimiento para su carga de trabajo.
- Optimizado: un grupo de escalado automático se considera optimizado cuando Compute Optimizer determina que el grupo está aprovisionado correctamente para ejecutar la carga de trabajo, según el tipo de instancia elegido. Para recursos optimizados, Compute Optimizer puede recomendar a veces un tipo de instancia de nueva generación.
- Ninguno: no hay recomendaciones para este grupo de escalado automático. Esto puede ocurrir si hace menos de 12 horas que se dio de alta en Compute Optimizer, cuando el grupo de escalado automático lleva ejecutándose menos de 30 horas o cuando el grupo de escalado automático o el tipo de instancia no es compatible con Compute Optimizer. Para obtener más información, consulte la sección Limitaciones.

Ver recomendaciones

Una vez que haya optado por Compute Optimizer, puede ver las conclusiones y recomendaciones que genera para los grupos de Auto Scaling. Si ha optado recientemente, es posible que las recomendaciones no estén disponibles durante un máximo de 12 horas.

Para ver las recomendaciones generadas para un grupo de escalado automático

1. Abre la consola de Compute Optimizer en. https://console.aws.amazon.com/compute-optimizer/

   Se abrirá la página de Panel.
2. Elija View recommendations for all Auto Scaling groups (Ver recomendaciones para todos los grupos de Auto Scaling).
3. Seleccione el grupo de escalado automático.

4. Elija View detail (Ver detalles).

   La vista cambia para mostrar hasta tres recomendaciones de instancia diferentes en una vista preconfigurada, en función de la configuración predeterminada de la tabla. También proporciona datos de CloudWatch métricas recientes (uso promedio de la CPU, promedio de entrada de red y promedio de salida de red) para el grupo Auto Scaling.

Determine si desea utilizar alguna de las recomendaciones. Decida si desea optimizar para mejora del rendimiento, para reducción de costos o para una combinación de ambos.

Para cambiar el tipo de instancia del grupo de escalado automático, actualice la plantilla de lanzamiento o el grupo de escalado automático para que utilicen una nueva configuración de lanzamiento. Las instancias existentes siguen utilizando la configuración anterior. Para actualizar las instancias existentes, termínelas de forma que se sustituyan por el grupo de escalado automático o permita que el escalado automático reemplace gradualmente las instancias más antiguas por instancias más recientes en función de sus políticas de terminación.

> **Note**
>
> Con la duración máxima de la instancia y las características de actualización de instancias, también puede reemplazar las instancias existentes de su grupo de escalado automático para iniciar nuevas instancias que utilicen la nueva plantilla de lanzamiento o configuración de lanzamiento. Para obtener más información, consulte Reemplazo de instancias de Auto Scaling en función de la duración máxima de la instancia y Uso de una actualización de instancias para actualizar instancias en un grupo de escalado automático.

Consideraciones para evaluar las recomendaciones

Antes de pasar a un nuevo tipo de instancia, tenga en cuenta lo siguiente:

- Las recomendaciones no prevén el uso que hará de ellas. Las recomendaciones se basan en su uso histórico durante el periodo de 14 días más reciente. Asegúrese de elegir un tipo de instancia que se espera que satisfaga sus necesidades de uso futuras.
- Céntrese en las métricas gráficas para determinar si el uso real es inferior a la capacidad de la instancia. También puede ver los datos métricos (promedio, pico, percentil) para evaluar más CloudWatch a fondo sus recomendaciones de instancias de EC2. Por ejemplo, observe cómo cambian las métricas de porcentaje de CPU durante el día y si hay picos que deben

acomodarse. Para obtener más información, consulta Cómo ver las métricas disponibles en la Guía del CloudWatch usuario de Amazon.

- Compute Optimizer podría proporcionar recomendaciones para instancias de rendimiento ampliable, que son las instancias T3, T3a y T2. Si superas periódicamente tu valor de referencia, asegúrate de poder seguir haciéndolo en función de la v CPUs del nuevo tipo de instancia. Para obtener más información, consulte Créditos de CPU y rendimiento de referencia para las instancias de rendimiento ampliable en la Guía del usuario de Amazon EC2.
- Si ha comprado una Instancia reservada, es posible que su instancia a petición se le facture como una Instancia reservada. Antes de cambiar el tipo de instancia actual, evalúe primero el impacto en la utilización y la cobertura de la instancia reservada.
- Considere la posibilidad de cambiar a instancias de nueva generación, siempre que sea posible.
- Al migrar a una familia de instancias diferente, asegúrese de que el tipo de instancia actual y el nuevo tipo de instancia sean compatibles, por ejemplo, en cuanto a virtualización, arquitectura o tipo de red. Para obtener más información, consulte Compatibilidad con el redimensionamiento de instancias en la Guía del usuario de Amazon EC2.
- Por último, considere la calificación de riesgo de rendimiento que se proporciona para cada recomendación. El riesgo de rendimiento indica la cantidad de esfuerzo que puede necesitar invertir para validar si el tipo de instancia recomendado cumple los requisitos de rendimiento de la carga de trabajo. También es recomendable que realice pruebas de carga y rendimiento rigurosas antes y después de realizar cualquier cambio.

Recursos adicionales

Además de los temas de esta página, consulte los siguientes recursos:

- Tipos de instancias de Amazon EC2
- AWS Compute Optimizer Guía del usuario

Utilice Elastic Load Balancing para distribuir el tráfico entrante de la aplicación en su grupo de escalado automático.

Elastic Load Balancing distribuye automáticamente el tráfico entrante de la aplicación entre todas las instancias EC2 que están en ejecución. Elastic Load Balancing ayuda a administrar las solicitudes entrantes dirigiendo el tráfico de manera óptima para que ninguna instancia supere su capacidad. Para utilizar Elastic Load Balancing con el grupo de Auto Scaling, asocie el balanceador de carga

al grupo de Auto Scaling. De este modo, se registra el grupo en el balanceador de carga, que actúa como un único punto de contacto para todo el tráfico web entrante al grupo de Auto Scaling.

Si utiliza Elastic Load Balancing con el grupo de Auto Scaling, no es necesario registrar las instancias EC2 individuales con el balanceador de carga. Las instancias lanzadas por el grupo de Auto Scaling se registran automáticamente en el balanceador de carga. Del mismo modo, se anula automáticamente el registro en el balanceador de carga de las instancias que el grupo de Auto Scaling termina.

Después de adjuntar un balanceador de carga al grupo de Auto Scaling, puede configurar el grupo de Auto Scaling para que utilice métricas de Elastic Load Balancing (como el recuento de solicitudes del Application Load Balancer por destino) para escalar el número de instancias del grupo a medida que fluctúe la demanda.

Opcionalmente, puede agregar comprobaciones de estado de Elastic Load Balancing al grupo de Auto Scaling para que Amazon EC2 Auto Scaling pueda identificar y reemplazar instancias que no estén en buen estado en función de estas comprobaciones de estado adicionales. De lo contrario, puede crear una CloudWatch alarma que le notifique si el número de anfitriones en buen estado del grupo objetivo es inferior al permitido.

Contenido

- Tipos de Elastic Load Balancing
- Preparación para asociar un equilibrador de carga de Elastic Load Balancing
- Asociación de un equilibrador de carga de Elastic Load Balancing al grupo de escalado automático
- Configuración de un equilibrador de carga de aplicación o un equilibrador de carga de red desde la consola
- Verificación del estado de asociación del equilibrador de carga
- Agregar una zona de disponibilidad
- Eliminar una zona de disponibilidad
- Desasociación de un grupo de destino o equilibrador de carga clásico de su grupo de escalado automático
- Ejemplos de cómo trabajar con Elastic Load Balancing mediante AWS CLI

Tipos de Elastic Load Balancing

Elastic Load Balancing ofrece cuatro tipos de balanceadores de carga que se pueden utilizar con el grupo de Auto Scaling: balanceadores de carga de aplicaciones, balanceadores de carga de red, balanceadores de carga de gateway y balanceadores de carga clásicos.

Hay una diferencia clave en el modo en que se configuran los tipos de equilibrador de carga. Con los balanceadores de carga de aplicaciones, los balanceadores de carga de red y los balanceadores de carga de gateway, las instancias se registran como destinos en un grupo de destino y puede dirigir el tráfico al grupo de destino. Con los balanceadores de carga clásicos, las instancias se registran directamente en el balanceador de carga.

Equilibrador de carga de aplicación

Enruta y balancea la carga en la capa de la aplicación (HTTP/HTTPS) y admite el enrutamiento basado en rutas. Un Application Load Balancer puede dirigir las solicitudes a puertos de uno o varios destinos registrados, como instancias EC2, en la nube virtual privada (VPC).

Equilibrador de carga de red

Dirige y equilibra la carga en la capa de transporte (capa 4 de TCP/UDP) basándose en la información de las direcciones que extrae del encabezado de la capa 4. Los balanceadores de carga de red pueden gestionar ráfagas de tráfico, conservar la IP de origen del cliente y utilizar una IP fija mientras dura la vida útil del balanceador de carga.

Balanceador de carga de gateway

Distribuye el tráfico a una flota de instancias de dispositivo. Proporciona escalabilidad, disponibilidad y simplicidad para dispositivos virtuales de terceros, como firewalls, sistemas de prevención y detección de intrusiones y otros dispositivos. Los balanceadores de carga de gateway funcionan con dispositivos virtuales compatibles con el protocolo GENEVE. Se requiere una integración técnica adicional, así que asegúrese de consultar la guía del usuario antes de elegir un balanceador de carga de gateway.

Equilibrador de carga clásico

Las rutas y los equilibrios de carga se encuentran en la capa de transporte (TCP/SSL), or at the application layer (HTTP/HTTPS).

Para conocer mejor los diferentes tipos de equilibradores de carga disponibles, consulte los siguientes recursos:

- ¿Qué es Elastic Load Balancing?
- ¿Qué es un Application Load Balancer?
- ¿Qué es un Network Load Balancer?
- ¿Qué es un balanceador de carga de gateway?
- ¿Qué es un Classic Load Balancer?

Preparación para asociar un equilibrador de carga de Elastic Load Balancing

Debe cumplir estos requisitos previos antes de conectar un equilibrador de carga de Elastic Load Balancing al grupo de escalado automático:

- Debe haber creado ya el equilibrador de carga y el grupo de destino que se usa para dirigir el tráfico al grupo de escalado automático.

  Hay dos formas de crear el equilibrador de carga y el grupo de destino:

  - Uso de Elastic Load Balancing: siga los procedimientos de la documentación de Elastic Load Balancing para crear y configurar el equilibrador de carga y el grupo de destino antes de crear el grupo de escalado automático. Omita el paso para registrar las instancias de Amazon EC2. Amazon EC2 Auto Scaling se encarga de registrar de manera automática las instancias, así como de anular dicho registro, cuando conecta un grupo de destino al grupo de escalado automático. Para obtener más información, consulte Introducción a Elastic Load Balancing en la Guía del usuario de Elastic Load Balancing.
  - Uso de Amazon EC2 Auto Scaling: crear, configurar y asociar el equilibrador de carga y el grupo de destino con una configuración básica desde la consola de Amazon EC2 Auto Scaling. Para obtener más información, consulte Configuración de un equilibrador de carga de aplicación o un equilibrador de carga de red desde la consola.
- Antes de crear un equilibrador de carga, conozca cuál necesita. Para obtener más información, consulte Tipos de Elastic Load Balancing.
- El balanceador de cargas y su grupo objetivo deben estar en la misma Cuenta de AWS VPC y región que tu grupo de Auto Scaling.
- Los grupos de destino deben especificar el tipo de destino `instance`. No puede especificar un tipo de destino `ip` cuando se utiliza un grupo de Auto Scaling.
- Si su plantilla de inicialización para su grupo de escalado automático no contiene el grupo de seguridad correcto que permita que el tráfico entrante del equilibrador de carga, debe actualizar

la plantilla de inicialización. Las reglas recomendadas dependen del tipo de balanceador de carga y los tipos de backends que utilice el balanceador de carga. Por ejemplo, para dirigir el tráfico a los servidores web, permita el acceso HTTP entrante en el puerto 80 desde el balanceador de carga. Las instancias existentes no se actualizan con la nueva configuración cuando se modifica la plantilla de lanzamiento. Para actualizar las instancias existentes, puede comenzar una actualización de instancias para reemplazarlas. Para obtener más información, consulte Uso de una actualización de instancias para actualizar instancias en un grupo de escalado automático.

- Los grupos de seguridad en la plantilla de inicialización deben permitir el acceso desde el equilibrador de carga en el puerto correcto para que Elastic Load Balancing realice las comprobaciones de estado.
- Al implementar dispositivos virtuales detrás de un equilibrador de carga de puertas de enlace, la imagen de máquina de Amazon (AMI) en la plantilla de inicialización debe especificar el ID de una AMI que admita el protocolo GENEVE para permitir al grupo de escalado automático intercambiar tráfico con un equilibrador de carga de puertas de enlace. Además, los grupos de seguridad que especifique en la plantilla de inicialización deben permitir el tráfico UDP en el puerto 6081.

> **Tip**
>
> Si tiene scripts de arranque que tardan en completarse, de manera opcional puede agregar un enlace de ciclo de vida de lanzamiento a su grupo de escalado automático para retrasar el registro de las instancias detrás del equilibrador de carga antes de que sus scripts de arranque se hayan completado correctamente y que las aplicaciones de las instancias estén listas para aceptar tráfico. No puede agregar un enlace de ciclo de vida cuando crea inicialmente un grupo de Auto Scaling en la consola de Amazon EC2 Auto Scaling. Sin embargo, puede agregar un enlace de ciclo de vida después de crear el grupo. Para obtener más información, consulte Enlaces de ciclo de vida de Amazon EC2 Auto Scaling.

Configuración de las comprobaciones de estado de los destinos

Puede configurar comprobaciones de estado para sus objetivos registrados con un equilibrador de carga de Elastic Load Balancing para garantizar que puedan administrar el tráfico de forma adecuada. Los pasos específicos varían según el tipo de equilibrador de carga que utilice. Para obtener más información, consulte los siguientes recursos:

- Equilibradores de carga de aplicación. Consulte Comprobaciones de estado de los grupos de destino en la Guía del usuario para equilibradores de carga de aplicación.

- Equilibradores de carga de red. Consulte Comprobaciones de estado de los grupos de destino en la Guía del usuario de los equilibradores de carga de red.
- Equilibrador de carga de la puerta de enlace. Consulte Comprobaciones de estado de los grupos de destino en la Guía del usuario de los equilibradores de carga de la puerta de enlace.
- Equilibrador de carga clásico. Consulte Configuración de comprobaciones de estado para el balanceador de carga clásico en la Guía del usuario para Equilibradores de carga clásicos.

De forma predeterminada, Amazon EC2 Auto Scaling no considera que una instancia está en mal estado ni la reemplaza si no supera las comprobaciones de estado de Elastic Load Balancing. Las comprobaciones de estado predeterminadas de un grupo de escalado automático son solo comprobaciones de estado de EC2. Para obtener más información, consulte Comprobaciones de estado para instancias en un grupo de escalado automático.

Para permitir que Amazon EC2 Auto Scaling sustituya las instancias que Elastic Load Balancing considera que están en mal estado, puede configurar el grupo de escalado automático para que utilice las comprobaciones de estado de Elastic Load Balancing. De este modo, Amazon EC2 Auto Scaling considera que la instancia está en mal estado si no supera las comprobaciones de estado de EC2 o las comprobaciones de estado de Elastic Load Balancing. Si asocia varios grupos de destino del balanceador de carga o balanceadores de carga clásicos al grupo, todos ellos deben registrar la instancia como correcta para que se considere que está en buen estado. Si cualquiera de ellos informa de una instancia como en mal estado, el grupo de Auto Scaling reemplaza la instancia, aunque otros informen que es correcta.

Para obtener más información sobre cómo habilitar las comprobaciones de estado para su grupo de escalado automático, consulte Asociación de un equilibrador de carga de Elastic Load Balancing al grupo de escalado automático.

> **Note**
>
> Para asegurarse de que estas comprobaciones de estado comiencen lo antes posible, asegúrese de que el periodo de gracia de las comprobaciones de estado del grupo no sea demasiado alto, sino lo suficiente para que las comprobaciones de estado de Elastic Load Balancing puedan determinar si hay un objetivo disponible para gestionar las solicitudes. Para obtener más información, consulte Establezca el periodo de gracia de la comprobación de estado para un grupo de escalado automático.

Asociación de un equilibrador de carga de Elastic Load Balancing al grupo de escalado automático

En este tema, se describe cómo asociar un equilibrador de carga de Elastic Load Balancing al grupo de escalado automático. También describe cómo activar las comprobaciones de estado de Elastic Load Balancing para permitir que Amazon EC2 Auto Scaling sustituya las instancias que Elastic Load Balancing informa que están en mal estado.

De forma predeterminada, Amazon EC2 Auto Scaling solo reemplaza las instancias en mal estado o inaccesibles en función de las comprobaciones de estado de Amazon EC2. Si activa las comprobaciones de estado de Elastic Load Balancing, Amazon EC2 Auto Scaling puede reemplazar una instancia en ejecución si alguno de los equilibradores de carga de Elastic Load Balancing que asocie al grupo de escalado automático informa que está en mal estado.

Para obtener un tutorial sobre cómo adjuntar un equilibrador de carga de aplicación al grupo de escalado automático, consulte Tutorial: Configuración de una aplicación con escalado y balanceo de carga aplicados.

> **Important**
>
> Antes de continuar, complete todos los requisitos previos de la sección anterior.

Contenido

- Asociación de un grupo de destino o equilibrador de carga clásico
- Desasociación de un grupo de destino o equilibrador de carga clásico

Asociación de un grupo de destino o equilibrador de carga clásico

Si crea o actualiza un grupo de escalado automático, puede asociar uno o varios grupos de destino o equilibradores de carga clásicos. Al asociar un equilibrador de carga de aplicación, un equilibrador de carga de red o un equilibrador de carga de puerta de enlace, se asocia un grupo de destino en lugar del propio equilibrador de carga.

Siga los pasos de esta sección para utilizar la consola a fin de hacer lo siguiente:

- Asociación de un grupo de destino o un equilibrador de carga clásico a un grupo de escalado automático

- Activación de las comprobaciones de estado de Elastic Load Balancing

Para asociar un balanceador de carga existente mientras crea un nuevo grupo de Auto Scaling

1. Abra la consola Amazon EC2 en https://console.aws.amazon.com/ec2/y seleccione Auto Scaling Groups en el panel de navegación.
2. En la barra de navegación de la parte superior de la pantalla, seleccione la Región de AWS en la que creó su equilibrador de carga.
3. Elija Create Auto Scaling group (Crear grupo de escalado automático).
4. En los pasos 1 y 2, elija las opciones que desee y continúe en Paso 3: Configurar opciones avanzadas.
5. En Load balancing (Balanceo de carga), elija Attach to an existing load balancer (Enlazar a un balanceador de carga existente).
6. En Attach to an existing load balancer (Enlazar a un balanceador de carga existente), lleve a cabo una de las siguientes operaciones:

   a. En los balanceadores de carga de aplicaciones, los balanceadores de carga de red y los balanceadores de carga de gateway:

      Seleccione Choose from your load balancer target groups (Elegir entre los grupos de destino del balanceador de carga) y, a continuación, elija un grupo de destino en el campo Existing load balancer target groups (Grupos de destino existentes del balanceador de carga).

   b. En los balanceadores de carga clásicos:

      Seleccione Choose from Classic Load Balancers (Elegir entre los balanceadores de carga clásicos) y, a continuación, elija el balanceador de carga en el campo Classic Load Balancers (Balanceadores de carga clásicos).
7. (Opcional) En Comprobaciones de estado, Tipos de comprobaciones de estado adicionales, seleccione Activar las comprobaciones de estado de Elastic Load Balancing.
8. (Opcional) En Período de gracia de comprobación de estado, ingrese el tiempo, en segundos. Este es el tiempo que Amazon EC2 Auto Scaling debe esperar antes de comprobar el estado de una instancia una vez que pasa al estado `InService`. Para obtener más información, consulte Establezca el periodo de gracia de la comprobación de estado para un grupo de escalado automático.

9. Proceda a crear el grupo de Auto Scaling. Las instancias se registrarán automáticamente en el balanceador de carga una vez creado el grupo de Auto Scaling.

Asociación de un equilibrador de carga existente a su grupo de escalado automático luego de haberlo creado

1. Abra la consola Amazon EC2 en https://console.aws.amazon.com/ec2/y seleccione Auto Scaling Groups en el panel de navegación.
2. Seleccione la casilla situada junto al grupo de escalado automático.

   Se abre un panel dividido en la parte inferior de la página Auto Scaling groups (Grupos de Auto Scaling).
3. En la pestaña Integraciones, elija Equilibrador de carga y luego Editar.
4. En Load balancing (Balance de carga), realice una de las siguientes acciones:

   a. En Application, Network or Gateway Load Balancer target groups (Grupos de destino del Application Load Balancer, red o gateway), seleccione su casilla de verificación y elija un grupo de destino.

   b. En Classic Load Balancers (Balanceadores de carga clásicos), seleccione su casilla de verificación y elija el balanceador de carga.
5. Elija Actualizar.

Cuando termine de asociar el equilibrador de carga, si lo desea, puede activar las comprobaciones de estado que lo utilizan.

Cómo activar las comprobaciones de estado de Elastic Load Balancing

1. En la pestaña Details (Detalles), elija Health checks (Comprobaciones de estado), Edit (Editar).
2. En Comprobaciones de estado, Tipos de comprobaciones de estado adicionales, seleccione Activar las comprobaciones de estado de Elastic Load Balancing.
3. En Período de gracia de comprobación de estado, ingrese el tiempo, en segundos. Este es el tiempo que Amazon EC2 Auto Scaling debe esperar antes de comprobar el estado de una instancia una vez que pasa al estado `InService`. Para obtener más información, consulte Establezca el periodo de gracia de la comprobación de estado para un grupo de escalado automático.
4. Elija Actualizar.

> **Note**
>
> Puede usar AWS CLI para supervisar el estado del equilibrador de carga mientras está conectado. Cuando Amazon EC2 Auto Scaling registre correctamente las instancias y al menos una de ellas supere las comprobaciones de estado, recibirá el estado `InService`. Para obtener más información, consulte Verificación del estado de asociación del equilibrador de carga.

Desasociación de un grupo de destino o equilibrador de carga clásico

Cuando ya no necesite el balanceador de carga, utilice el siguiente procedimiento para desconectarlo del grupo de Auto Scaling.

Para desasociar un balanceador de carga de un grupo

1. Abra la consola Amazon EC2 en https://console.aws.amazon.com/ec2/y seleccione Auto Scaling Groups en el panel de navegación.
2. Seleccione la casilla de verificación situada junto a un grupo existente.

   Se abre un panel dividido en la parte inferior de la página Auto Scaling groups (Grupos de Auto Scaling).
3. En la pestaña Details (Detalles), elija Load balancing (Balance de carga), Edit (Editar).
4. En Load balancing (Balance de carga), realice una de las siguientes acciones:

   a. En Application, Network or Gateway Load Balancer target groups (Grupos de destino del Application Load Balancer, red o gateway), elija el icono de eliminación (X) situado junto al grupo de destino.

   b. En Classic Load Balancers (Balanceadores de carga clásicos), elija el icono de eliminación (X) situado junto al balanceador de carga.
5. Elija Actualizar.

Cuando termine de desasociar el grupo de destino, podrá desactivar las comprobaciones de estado de Elastic Load Balancing.

Cómo desactivar comprobaciones de estado de Elastic Load Balancing

1. En la pestaña Details (Detalles), elija Health checks (Comprobaciones de estado), Edit (Editar).

2. En Comprobaciones de estado, Tipos de comprobaciones de estado adicionales, anule la selección Activar las comprobaciones de estado de Elastic Load Balancing.
3. Elija Actualizar.

Configuración de un equilibrador de carga de aplicación o un equilibrador de carga de red desde la consola

Utilice el siguiente procedimiento para crear y asociar un Application Load Balancer o un Network Load Balancer a medida que crea el grupo de Auto Scaling.

Para cear y asociar un balanceador de carga existente mientras crea un nuevo grupo de Auto Scaling

1. Abra la consola Amazon EC2 en https://console.aws.amazon.com/ec2/y seleccione Auto Scaling Groups en el panel de navegación.
2. Elija Create Auto Scaling group (Crear grupo de escalado automático).
3. En los pasos 1 y 2, elija las opciones que desee y continúe en Paso 3: Configurar opciones avanzadas.
4. En Load balancing (Balanceo de carga), elija Attach to a new load balancer (Asociar a un nuevo balanceador de carga).
    a. En Attach to a new load balancer (Asociar a un nuevo balanceador de carga), en Load balancer type (Tipo de balanceador de carga), elija si desea crear un Application Load Balancer o un Network Load Balancer.
    b. En Load balancer name (Nombre del balanceador de carga), escriba un nombre para el balanceador de carga o conserve el nombre predeterminado.
    c. En Load balancer scheme (Esquema del balanceador de carga), elija si desea crear un balanceador de carga público orientado a Internet o si mantiene el valor predeterminado para un balanceador de carga interno.
    d. En Availability Zones and subnets (Zonas de disponibilidad y subredes), seleccione la subred pública de cada zona de disponibilidad en la que elija lanzar las instancias EC2. (Se rellenan previamente a partir del paso 2).
    e. En Listeners and routing (Agentes de escucha y enrutamiento, actualice el número de puerto de su agente de escucha (si es necesario) y en Default routing (Enrutamiento

predeterminado), elija Create a target group (Crear un grupo de destino). O bien, puede elegir un grupo de destino existente en la lista desplegable.

f. Si eligió Create a target group (Crear un grupo de destino) en el último paso, en New target group name (Nombre del nuevo grupo de destino), escriba un nombre para el grupo de destino o conserve el nombre predeterminado.

g. Para agregar etiquetas al equilibrador de carga, elija Add Tags (Agregar etiquetas), facilite una clave y un valor para cada etiqueta.

5. (Opcional) En Comprobaciones de estado, Tipos de comprobaciones de estado adicionales, seleccione Activar las comprobaciones de estado de Elastic Load Balancing.

6. (Opcional) En Período de gracia de comprobación de estado, ingrese el tiempo, en segundos. Este es el tiempo que Amazon EC2 Auto Scaling debe esperar antes de comprobar el estado de una instancia una vez que pasa al estado `InService`. Para obtener más información, consulte Establezca el periodo de gracia de la comprobación de estado para un grupo de escalado automático.

7. Proceda a crear el grupo de Auto Scaling. Las instancias se registrarán automáticamente en el balanceador de carga una vez creado el grupo de Auto Scaling.

> **Note**
>
> Después de crear el grupo de Auto Scaling, puede utilizar la consola de Elastic Load Balancing para crear agentes de escucha adicionales. Esto resulta útil si necesita crear un agente de escucha con un protocolo seguro, como HTTPS o un agente de escucha UDP. Puede agregar más agentes de escucha a los balanceadores de carga existentes, siempre y cuando utilice puertos distintos.

Verificación del estado de asociación del equilibrador de carga

Después de asociar un equilibrador de carga, este pasa a tener el estado `Adding` mientras registra las instancias del grupo. Cuando se registran todas las instancias del grupo, entra en el estado `Added`. Cuando al menos una de las instancias registradas supera las comprobaciones de estado, pasa a tener el estado `InService`. Cuando el balanceador de carga se encuentra en el estado `InService`, Amazon EC2 Auto Scaling puede terminar y reemplazar las instancias notificadas como en mal estado. Si ninguna de las instancias registradas supera las comprobaciones de estado (debido, por ejemplo, a una comprobación de estado configurada incorrectamente), el balanceador

de carga no pasa al estado `InService`. Amazon EC2 Auto Scaling no termina y reemplaza las instancias.

Cuando desasocia un balanceador de carga, este pasa a tener el estado `Removing` mientras se cancela el registro de las instancias del grupo. Las instancias siguen ejecutándose una vez que se cancela el registro. De forma predeterminada, el drenaje de conexión (retardo de anulación del registro) está habilitado para Application Load Balancers, Network Load Balancers y Gateway Load Balancers. Si Connection Draining está habilitado, Elastic Load Balancing espera a que se completen las solicitudes en tránsito o a que termine el tiempo de espera máximo (lo que ocurra primero) antes de cancelar el registro de las instancias.

Puede verificar el estado del archivo adjunto mediante AWS Command Line Interface (AWS CLI) o AWS SDKs. No puede verificar el estado de asociación desde la consola.

Para usar el AWS CLI para verificar el estado del archivo adjunto

El siguiente describe-traffic-sourcescomando devuelve el estado de los adjuntos de todas las fuentes de tráfico del grupo de Auto Scaling especificado.

```
aws autoscaling describe-traffic-sources --auto-scaling-group-name my-asg
```

El ejemplo devuelve el ARN del grupo de destino de Elastic Load Balancing que está asociado al grupo de escalado automático, junto con el estado de asociación del grupo de destino en el elemento `State`.

```
{
    "TrafficSources": [
        {
            "Identifier": "arn:aws:elasticloadbalancing:region:account-
id:targetgroup/my-targets/1234567890123456",
            "State": "InService",
            "Type": "elbv2"
        }
    ]
}
```

Agregar una zona de disponibilidad

Para aprovecharse de la seguridad y la fiabilidad de la redundancia geográfica, distribuya el grupo de escalado automático entre varias zonas de disponibilidad de una región en la que trabaje y asocie después un equilibrador de carga para distribuir el tráfico entrante entre las zonas de disponibilidad.

Cuando una zona de disponibilidad pasa a tener un estado incorrecto o deja de estar disponible, Amazon EC2 Auto Scaling lanza nuevas instancias en la zona de disponibilidad afectada. Cuando la zona de disponibilidad en mal estado vuelve a tener un estado correcto, Amazon EC2 Auto Scaling redistribuye automáticamente las instancias de la aplicación de manera uniforme por todas las zonas de disponibilidad del grupo de Auto Scaling. Para ello, Amazon EC2 Auto Scaling intenta lanzar nuevas instancias en la zona de disponibilidad con el menor número de instancias. Sin embargo, el intento fracasa, Amazon EC2 Auto Scaling intenta lanzar instancias en otras zonas de disponibilidad hasta que lo consiga.

Elastic Load Balancing crea un nodo de balanceador de carga para cada zona de disponibilidad que habilita para el balanceador de carga. Si habilita el balanceo de carga entre zonas para el balanceador de carga, cada nodo del balanceador de carga distribuye el tráfico equitativamente entre las instancias registradas en todas las zonas de disponibilidad habilitadas. Si el equilibrio de cargas entre zonas está inhabilitado, cada nodo del equilibrador de carga distribuye las solicitudes equitativamente entre todas las instancias registradas solo en su zona de disponibilidad.

Deberá especificar al menos una zona de disponibilidad cuando crea el grupo de Auto Scaling. Después, puede ampliar la disponibilidad de su aplicación agregando una zona de disponibilidad al grupo de Auto Scaling y habilitando esa zona de disponibilidad para el balanceador de carga (si este lo admite).

Limitaciones

Para actualizar las zonas de disponibilidad que están habilitadas para el balanceador de carga, debe conocer las siguientes limitaciones:

- Cuando se habilita una zona de disponibilidad para el balanceador de carga, se especifica una subred de esa zona de disponibilidad. Tenga en cuenta que puede habilitar como máximo una subred por cada zona de disponibilidad para el balanceador de carga.
- Para los balanceadores de carga expuestos a Internet, las subredes que especifique para el balanceador de carga deben tener al menos ocho direcciones IP disponibles.
- Para los balanceadores de carga de aplicaciones, debe habilitar al menos dos zonas de disponibilidad.

- En el caso de los balanceadores de carga de red, no puede desactivar las zonas de disponibilidad habilitadas, pero puede habilitar otras adicionales.
- En el caso de los equilibradores de carga de puerta de enlace, no puede desactivar las zonas de disponibilidad habilitadas, pero puede habilitar otras adicionales.

Utilice el siguiente procedimiento para ampliar el grupo de Auto Scaling y el balanceador de carga a una subred en una zona de disponibilidad adicional.

Para agregar una zona de disponibilidad

1. Abra la consola Amazon EC2 en https://console.aws.amazon.com/ec2/y seleccione Auto Scaling Groups en el panel de navegación.
2. Seleccione la casilla de verificación situada junto a un grupo existente.

   Se abre un panel dividido en la parte inferior de la página Auto Scaling groups (Grupos de Auto Scaling).
3. En la pestaña Details (Detalles), elija Network (Red), Edit (Editar).
4. En Subredes, elija la subred correspondiente a la zona de disponibilidad que desee agregar al grupo de Auto Scaling.
5. Elija Actualizar.
6. Para actualizar las zonas de disponibilidad del balanceador de carga para que comparta las mismas zonas de disponibilidad que el grupo de Auto Scaling, realice los pasos siguientes:

   a. En el panel de navegación, en Equilibrio de carga, elija Equilibradores de carga.

   b. Elija el equilibrador de carga de .

   c. Realice una de las siguientes acciones:

      - Para los balanceadores de carga de aplicaciones y los balanceadores de carga de red:

        1. En la pestaña Description (Descripción), en Availability Zones (Zonas de disponibilidad), elija Edit subnets (Editar las subredes).
        2. En la página Edit subnets (Editar las subredes), en Availability Zones (Zonas de disponibilidad), seleccione la casilla de verificación de la zona de disponibilidad que desea agregar. Si solo hay una subred para esa zona, se selecciona. Si hay más de una subred para esa zona, seleccione una de ellas.

      - Para balanceadores de carga clásicos en una VPC:

1. En la pestaña Instances (Instancias), elija Edit Availability Zones (Editar zonas de disponibilidad).
2. En la página Add and Remove Subnets (Agregar y eliminar subredes), en Available subnets (Subredes disponibles), seleccione la subred utilizando su icono de adición (+). La subred se situará bajo Selected subnets (Subredes seleccionadas).

d. Seleccione Save.

Recursos relacionados

Amazon EC2 Auto Scaling reequilibra el grupo al cambiar las zonas de disponibilidad. Esto implica reemplazar y redistribuir algunas instancias. Para obtener más información, consulte Ejemplo: distribuir instancias entre zonas de disponibilidad.

Si ha registrado destinos en zonas de disponibilidad que no están habilitadas para el equilibrador de cargas, este no dirige el tráfico hacia ellas. Para obtener más información, consulte Funcionamiento de Elastic Load Balancing en la Guía del usuario de Elastic Load Balancing.

Eliminar una zona de disponibilidad

Para quitar una zona de disponibilidad del grupo de Auto Scaling y el balanceador de carga, utilice el siguiente procedimiento.

Para eliminar una zona de disponibilidad

1. Abra la consola Amazon EC2 en https://console.aws.amazon.com/ec2/y seleccione Auto Scaling Groups en el panel de navegación.
2. Seleccione la casilla de verificación situada junto a un grupo existente.

   Se abre un panel dividido en la parte inferior de la página Auto Scaling groups (Grupos de Auto Scaling).
3. En la pestaña Details (Detalles), elija Network (Red), Edit (Editar).
4. En Subnets (Subredes), elija el icono de eliminación (X) de la subred correspondiente a la zona de disponibilidad que desee quitar del grupo de Auto Scaling. Si hay más de una subred en esa zona, elija el icono de eliminación (X) para cada una de ellas.
5. Elija Actualizar.

6. Para actualizar las zonas de disponibilidad del balanceador de carga para que comparta las mismas zonas de disponibilidad que el grupo de Auto Scaling, realice los pasos siguientes:

   a. En el panel de navegación, en Equilibrio de carga, elija Equilibradores de carga.

   b. Elija el equilibrador de carga de .

   c. Realice una de las siguientes acciones:

      - Para equilibradores de carga de aplicaciones:

        1. En la pestaña Description (Descripción), en Availability Zones (Zonas de disponibilidad), elija Edit subnets (Editar las subredes).
        2. En la página Edit subnets (Editar las subredes), en Availability Zones (Zonas de disponibilidad), borre la casilla de verificación para eliminar la subred de la zona de disponibilidad.

      - Para balanceadores de carga clásicos en una VPC:

        1. En la pestaña Instances (Instancias), elija Edit Availability Zones (Editar zonas de disponibilidad).
        2. En la página Add and Remove Subnets (Agregar y eliminar subredes), en Available subnets (Subredes disponibles), quite la subred utilizando su icono de eliminación (-). La subred se situará bajo Available Subnets (Subredes disponibles).

   d. Seleccione Save.

Desasociación de un grupo de destino o equilibrador de carga clásico de su grupo de escalado automático

Cuando ya no necesite el balanceador de carga, utilice el siguiente procedimiento para desconectarlo del grupo de Auto Scaling.

Para desasociar un balanceador de carga de un grupo

1. Abra la consola Amazon EC2 en https://console.aws.amazon.com/ec2/y seleccione Auto Scaling Groups en el panel de navegación.
2. Seleccione la casilla de verificación situada junto a un grupo existente.

   Se abre un panel dividido en la parte inferior de la página Auto Scaling groups (Grupos de Auto Scaling).

3. En la pestaña Details (Detalles), elija Load balancing (Balance de carga), Edit (Editar).
4. En Load balancing (Balance de carga), realice una de las siguientes acciones:
   a. En Application, Network or Gateway Load Balancer target groups (Grupos de destino del Application Load Balancer, red o gateway), elija el icono de eliminación (X) situado junto al grupo de destino.
   b. En Classic Load Balancers (Balanceadores de carga clásicos), elija el icono de eliminación (X) situado junto al balanceador de carga.
5. Elija Actualizar.

Cuando termine de desasociar el grupo de destino, podrá desactivar las comprobaciones de estado de Elastic Load Balancing.

Cómo desactivar comprobaciones de estado de Elastic Load Balancing

1. En la pestaña Details (Detalles), elija Health checks (Comprobaciones de estado), Edit (Editar).
2. En Comprobaciones de estado, Tipos de comprobaciones de estado adicionales, anule la selección Activar las comprobaciones de estado de Elastic Load Balancing.
3. Elija Actualizar.

Ejemplos de cómo trabajar con Elastic Load Balancing mediante AWS CLI

Utilice AWS Command Line Interface (AWS CLI) para adjuntar, separar y describir los balanceadores de carga y los grupos objetivo, añadir y eliminar comprobaciones de estado de Elastic Load Balancing y cambiar las zonas de disponibilidad que están habilitadas.

En este tema se muestran ejemplos de AWS CLI comandos que realizan tareas comunes para Amazon EC2 Auto Scaling.

> **Important**
>
> Para obtener más ejemplos de comandos, consulte `aws elbv2` y `aws elb` en la Referencia de los comandos de AWS CLI .

Contenido

- Asociación de su grupo de destino o equilibrador de carga clásico
- Descripción de sus grupos de destino o equilibradores de carga clásicos
- Adición de comprobaciones de estado de Elastic Load Balancing
- Cambio de sus zonas de disponibilidad
- Desasociación de su grupo de destino o equilibrador de carga clásico
- Eliminación de las comprobaciones de estado Elastic Load Balancing
- Comandos heredados

Asociación de su grupo de destino o equilibrador de carga clásico

Utilice el siguiente create-auto-scaling-groupcomando para crear un grupo de Auto Scaling y adjuntar simultáneamente un grupo de destino especificando su nombre de recurso de Amazon (ARN). El grupo de destino puede asociarse con un equilibrador de carga de aplicación, un equilibrador de carga de red o un equilibrador de carga de puerta de enlace.

Sustituya los valores de muestra de `--auto-scaling-group-name`, `--vpc-zone-identifier`, `--min-size` y `--max-size`. Para la opción `--launch-template`, sustituya *`my-launch-template`* y *`1`* por el nombre y la versión de una plantilla de lanzamiento para su grupo de escalado automático. Para la opción `--traffic-sources`, sustituya el ARN de muestra por el ARN de un grupo de destino para un equilibrador de carga de aplicación, un equilibrador de carga de red o un equilibrador de carga de puerta de enlace.

```
aws autoscaling create-auto-scaling-group --auto-scaling-group-name my-asg \
  --launch-template LaunchTemplateName=my-launch-template,Version='1' \
  --vpc-zone-identifier "subnet-5ea0c127,subnet-6194ea3b,subnet-c934b782" \
  --min-size 1 --max-size 5 \
  --traffic-sources "Identifier=arn:aws:elasticloadbalancing:region:account-
id:targetgroup/my-targets/12345678EXAMPLE1"
```

Use el attach-traffic-sourcescomando para adjuntar grupos de destino adicionales al grupo Auto Scaling una vez creado.

El siguiente comando agrega otro grupo de destino al mismo grupo.

```
aws autoscaling attach-traffic-sources --auto-scaling-group-name my-asg \
  --traffic-sources "Identifier=arn:aws:elasticloadbalancing:region:account-
id:targetgroup/my-targets/12345678EXAMPLE2"
```

Como alternativa, para asociar un equilibrador de carga clásico a su grupo, especifique las opciones `--traffic-sources` y `--type` cuando utilice create-auto-scaling-group o attach-traffic-sources, como en el siguiente ejemplo. Reemplace *my-classic-load-balancer* por el nombre de un equilibrador de carga clásico. Para la opción `--type`, especifique un valor de **elb**.

```
--traffic-sources "Identifier=my-classic-load-balancer" --type elb
```

Descripción de sus grupos de destino o equilibradores de carga clásicos

Para describir los balanceadores de carga o los grupos objetivo adjuntos a su grupo de Auto Scaling, utilice el siguiente describe-traffic-sourcescomando. Reemplace *my-asg* por el nombre de su grupo.

```
aws autoscaling describe-traffic-sources --auto-scaling-group-name my-asg
```

El ejemplo devuelve el ARN de los grupos de destino de Elastic Load Balancing que asoció al grupo de escalado automático.

```
{
    "TrafficSources": [
        {
            "Identifier": "arn:aws:elasticloadbalancing:region:account-
id:targetgroup/my-targets/12345678EXAMPLE1",
            "State": "InService",
            "Type": "elbv2"
        },
        {
            "Identifier": "arn:aws:elasticloadbalancing:region:account-
id:targetgroup/my-targets/12345678EXAMPLE2",
            "State": "InService",
            "Type": "elbv2"
        }
    ]
}
```

Para obtener una explicación del campo `State`, consulte Verificación del estado de asociación del equilibrador de carga.

Adición de comprobaciones de estado de Elastic Load Balancing

Para añadir las comprobaciones de estado de Elastic Load Balancing a las comprobaciones de estado que su grupo de Auto Scaling realiza en las instancias, utilice el siguiente update-auto-

scaling-groupcomando y especifique **ELB** el valor de la `--health-check-type` opción. Reemplace *my-asg* por el nombre de su grupo.

```
aws autoscaling update-auto-scaling-group --auto-scaling-group-name my-asg \
  --health-check-type "ELB"
```

Las instancias nuevas a menudo necesitan tiempo para una breve preparación antes de que puedan pasar una comprobación de estado. Si el período de gracia no proporciona suficiente tiempo de preparación, es posible que las instancias no aparezcan listas para servir tráfico. Amazon EC2 Auto Scaling podría considerar que esas instancias no están en buen estado y reemplazarlas.

Para actualizar el período de gracia de la comprobación de estado, utilice la opción `--health-check-grace-period` cuando use update-auto-scaling-group, como en el siguiente ejemplo. *300*Sustitúyalo por el número de segundos para mantener las nuevas instancias en servicio antes de cancelarlas si se descubre que no están en buen estado.

```
--health-check-grace-period 300
```

Para obtener más información, consulte Comprobaciones de estado para instancias en un grupo de escalado automático.

Cambio de sus zonas de disponibilidad

Cambiar las zonas de disponibilidad tiene algunas limitaciones que debe conocer. Para obtener más información, consulte Agregar una zona de disponibilidad.

Cambio de las zonas de disponibilidad de un equilibrador de carga de aplicación o un equilibrador de carga de red

1. Antes de cambiar las zonas de disponibilidad del equilibrador de cargas, se recomienda actualizar primero las zonas de disponibilidad del grupo de escalado automático para comprobar que hay disponibilidad para los tipos de instancia en las zonas especificadas.

   Para actualizar las zonas de disponibilidad de su grupo de Auto Scaling, utilice el siguiente update-auto-scaling-groupcomando. Sustituya la subred IDs de ejemplo por la IDs de las subredes de las zonas de disponibilidad para activarla. Las subredes especificadas sustituyen a las subredes habilitadas anteriormente. Reemplace *my-asg* por el nombre de su grupo.

   ```
   aws autoscaling update-auto-scaling-group --auto-scaling-group-name my-asg \
   ```

```
  --vpc-zone-identifier "subnet-41767929,subnet-cb663da2,subnet-8360a9e7"
```

2. Usa el siguiente describe-auto-scaling-groupscomando para verificar que se hayan lanzado las instancias de las nuevas subredes. Si las instancias se han lanzado, verá una lista de las instancias y sus estados. Reemplace *my-asg* por el nombre de su grupo.

```
aws autoscaling describe-auto-scaling-groups --auto-scaling-group-name my-asg
```

3. Use el siguiente comando set-subnets para especificar las subredes de su equilibrador de carga. Reemplace la subred IDs de ejemplo por la IDs de las subredes de las zonas de disponibilidad para habilitarla. Puede especificar solo una subred por zona de disponibilidad. Las subredes especificadas sustituyen a las subredes habilitadas anteriormente. Reemplace *my-lb-arn* por el ARN de su equilibrador de carga.

```
aws elbv2 set-subnets --load-balancer-arn my-lb-arn \
  --subnets subnet-41767929 subnet-cb663da2 subnet-8360a9e7
```

Cambio de las zonas de disponibilidad de un equilibrador de carga clásico

1. Antes de cambiar las zonas de disponibilidad del equilibrador de cargas, se recomienda actualizar primero las zonas de disponibilidad del grupo de escalado automático para comprobar que hay disponibilidad para los tipos de instancia en las zonas especificadas.

   Para actualizar las zonas de disponibilidad de su grupo de Auto Scaling, utilice el siguiente update-auto-scaling-groupcomando. Sustituya la subred IDs de ejemplo por la IDs de las subredes de las zonas de disponibilidad para activarla. Las subredes especificadas sustituyen a las subredes habilitadas anteriormente. Reemplace *my-asg* por el nombre de su grupo.

```
aws autoscaling update-auto-scaling-group --auto-scaling-group-name my-asg \
  --vpc-zone-identifier "subnet-41767929,subnet-cb663da2"
```

2. Usa el siguiente describe-auto-scaling-groupscomando para verificar que se hayan lanzado las instancias de las nuevas subredes. Si las instancias se han lanzado, verá una lista de las instancias y sus estados. Reemplace *my-asg* por el nombre de su grupo.

```
aws autoscaling describe-auto-scaling-groups --auto-scaling-group-name my-asg
```

3. Usa el siguiente comando attach-load-balancer-to-subnets para habilitar una nueva zona de disponibilidad para tu Classic Load Balancer. Sustituya el ID de la subred de muestra por el ID

de la subred de las zonas de disponibilidad para habilitarlas. Sustituya *my-lb* por el nombre de su equilibrador de carga.

```
aws elb attach-load-balancer-to-subnets --load-balancer-name my-lb \
  --subnets subnet-cb663da2
```

Para deshabilitar una zona de disponibilidad, usa el siguiente detach-load-balancer-from comando -subnets. Sustituya el ID de la subred de muestra por el ID de la subred de las zonas de disponibilidad para deshabilitarlas. Sustituya *my-lb* por el nombre de su equilibrador de carga.

```
aws elb detach-load-balancer-from-subnets --load-balancer-name my-lb \
  --subnets subnet-8360a9e7
```

Desasociación de su grupo de destino o equilibrador de carga clásico

El siguiente detach-traffic-sourcescomando separa un grupo objetivo de su grupo de Auto Scaling cuando ya no lo necesita.

Para la opción `--auto-scaling-group-name`, reemplace *my-asg* por el nombre de su grupo. Para la opción `--traffic-sources`, sustituya el ARN de muestra por el ARN de un grupo de destino para un equilibrador de carga de aplicación, un equilibrador de carga de red o un equilibrador de carga de puerta de enlace.

```
aws autoscaling detach-traffic-sources --auto-scaling-group-name my-asg \
  --traffic-sources "Identifier=arn:aws:elasticloadbalancing:region:account-
id:targetgroup/my-targets/1234567890123456"
```

Para separar un equilibrador de carga clásico de su grupo, especifique las opciones `--traffic-sources` y `--type`, como en el siguiente ejemplo. Reemplace *my-classic-load-balancer* por el nombre de un equilibrador de carga clásico. Para la opción `--type`, especifique un valor de **`elb`**.

```
--traffic-sources "Identifier=my-classic-load-balancer" --type elb
```

Eliminación de las comprobaciones de estado Elastic Load Balancing

Para eliminar las comprobaciones de estado de Elastic Load Balancing del grupo de Auto Scaling, utilice el siguiente update-auto-scaling-groupcomando y especifique **EC2** el valor de la `--health-check-type` opción. Reemplace *my-asg* por el nombre de su grupo.

```
aws autoscaling update-auto-scaling-group --auto-scaling-group-name my-asg \
  --health-check-type "EC2"
```

Para obtener más información, consulte Comprobaciones de estado para instancias en un grupo de escalado automático.

Comandos heredados

Los siguientes ejemplos muestran cómo usar comandos de CLI heredados para asociar, desasociar y describir equilibradores de carga y grupos de destino. Permanecen en este documento como referencia para cualquier cliente que quiera usarlos. Seguimos admitiendo los comandos CLI antiguos, pero le recomendamos que utilice los nuevos comandos CLI "fuentes de tráfico", que pueden asociar y desasociar varios tipos de fuentes de tráfico. Puede usar los comandos CLI heredados y los comandos CLI "fuentes de tráfico" en el mismo grupo de escalado automático.

Asociación de su grupo de destino o equilibrador de carga clásico (heredado)

Asociación de su grupo de destino

El siguiente create-auto-scaling-groupcomando crea un grupo de Auto Scaling con un grupo objetivo adjunto. Especifique el Nombre de recurso de Amazon (ARN) de un grupo de destino para un Application Load Balancer, un Network Load Balancer o un balanceador de carga de gateway.

```
aws autoscaling create-auto-scaling-group --auto-scaling-group-name my-asg \
  --launch-template LaunchTemplateName=my-launch-template,Version='1' \
  --vpc-zone-identifier "subnet-5ea0c127,subnet-6194ea3b,subnet-c934b782" \
  --target-group-arns "arn:aws:elasticloadbalancing:region:account-id:targetgroup/my-
targets/1234567890123456" \
  --min-size 1 --max-size 5
```

El siguiente comando attach-load-balancer-target-groups asocia un grupo objetivo a un grupo de Auto Scaling existente.

```
aws autoscaling attach-load-balancer-target-groups --auto-scaling-group-name my-asg \
```

```
  --target-group-arns "arn:aws:elasticloadbalancing:region:account-id:targetgroup/my-targets/1234567890123456"
```

Asociación de su equilibrador de carga clásico

El siguiente create-auto-scaling-groupcomando crea un grupo de Auto Scaling con un Classic Load Balancer adjunto.

```
aws autoscaling create-auto-scaling-group --auto-scaling-group-name my-asg \
  --launch-configuration-name my-launch-config \
  --vpc-zone-identifier "subnet-5ea0c127,subnet-6194ea3b,subnet-c934b782" \
  --load-balancer-names "my-load-balancer" \
  --min-size 1 --max-size 5
```

El siguiente attach-load-balancerscomando adjunta el Classic Load Balancer especificado a un grupo de Auto Scaling existente.

```
aws autoscaling attach-load-balancers --auto-scaling-group-name my-asg \
  --load-balancer-names my-lb
```

Descripción de su grupo de destino o equilibrador de carga clásico (heredado)

Descripción de grupos de destino

Para describir los grupos objetivo asociados a un grupo de Auto Scaling, utilice el comando describe-load-balancer-target-groups. En el siguiente ejemplo se enumeran los grupos objetivo de*my-asg*.

```
aws autoscaling describe-load-balancer-target-groups --auto-scaling-group-name my-asg
```

Descripción de los equilibradores de carga clásicos

Para describir los balanceadores de carga clásicos asociados a un grupo de Auto Scaling, usa el describe-load-balancerscomando. En el siguiente ejemplo, se enumeran los balanceadores de carga clásicos de. *my-asg*

```
aws autoscaling describe-load-balancers --auto-scaling-group-name my-asg
```

Desasociación de su grupo de destino o equilibrador de carga clásico (heredado)

Para desasociar un grupo de destino

El siguiente comando detach-load-balancer-target-groups separa un grupo objetivo del grupo de Auto Scaling cuando ya no lo necesita.

```
aws autoscaling detach-load-balancer-target-groups --auto-scaling-group-name my-asg \
  --target-group-arns "arn:aws:elasticloadbalancing:region:account-id:targetgroup/my-targets/1234567890123456"
```

Desasociación de un equilibrador de carga clásico

El siguiente detach-load-balancerscomando desconecta un Classic Load Balancer del grupo de Auto Scaling cuando ya no lo necesita.

```
aws autoscaling detach-load-balancers --auto-scaling-group-name my-asg \
  --load-balancer-names my-lb
```

Administración del flujo de tráfico con un grupo de destino de VPC Lattice

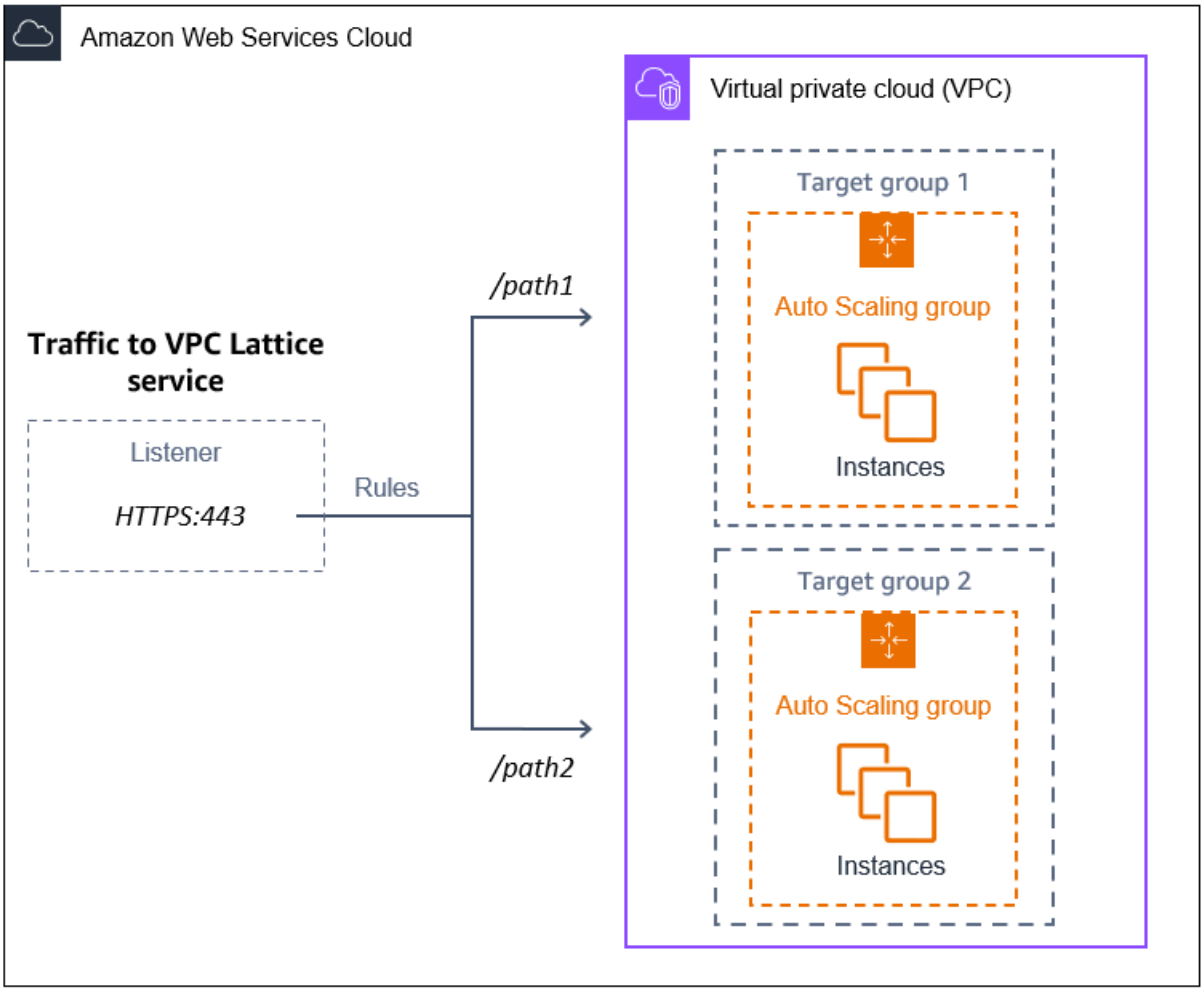
Puede usar Amazon VPC Lattice para administrar el flujo de tráfico y las llamadas a la API entre las aplicaciones y los servicios que se ejecutan en recursos independientes, como los grupos de escalado automático o las funciones de Lambda. VPC Lattice es un servicio de redes de aplicaciones que le permite conectar, proteger y monitorear todos sus servicios en varias cuentas y nubes privadas virtuales (VPC). Para obtener más información sobre VPC Lattice, consulte ¿Qué es VPC Lattice?

Para empezar a usar VPC Lattice, primero cree los recursos de VPC Lattice necesarios que permitan que los recursos de una VPC asociada a una red de servicios se conecten entre sí. Estos recursos incluyen los servicios, los oyentes, las reglas de oyente y los grupos de destino.

Para asociar un grupo de escalado automático a un servicio de VPC Lattice, cree un grupo de destino para el servicio que envíe las solicitudes a las instancias registradas por ID de instancia y agregue un oyente al servicio que envíe las solicitudes al grupo de destino. Después asocie el grupo de destino con su grupo de escalado automático. Amazon EC2 Auto Scaling registra automáticamente las instancias EC2 como destinos en el grupo de destino. Más adelante, cuando Amazon EC2 Auto Scaling necesite terminar una instancia, cancelará automáticamente el registro de la instancia en el grupo de destino antes de la terminación.

Después de asociar el grupo de destino, es el punto de entrada para todas las solicitudes entrantes a su grupo de escalado automático. Como se muestra en el ejemplo del siguiente diagrama, las

solicitudes entrantes se pueden dirigir al grupo de destino correspondiente mediante las reglas de oyente especificadas para un servicio de VPC Lattice.



Cuando el tráfico se dirige a través de VPC Lattice a su grupo de escalado automático, VPC Lattice equilibra las solicitudes entre las instancias del grupo mediante el equilibrio de carga por turnos. VPC Lattice también puede monitorizar el estado de las instancias registradas y dirigir el tráfico solo a las instancias en buen estado.

Para mantener las instancias disponibles para las solicitudes entrantes, si lo desea, puede añadir comprobaciones de estado de VPC Lattice a su grupo de escalado automático. De esta forma, si una de las instancias de EC2 experimenta un error, su grupo de escalado automático lanza automáticamente una instancia nueva para sustituirla. El comportamiento de las comprobaciones de estado de VPC Lattice es similar al comportamiento de las comprobaciones de estado de Elastic Load Balancing. Las comprobaciones de estado predeterminadas de un grupo de escalado automático son solo comprobaciones de estado de EC2.

Para obtener más información sobre VPC Lattice, consulte Simplifique la Service-to-Service conectividad, la seguridad y la supervisión con Amazon VPC Lattice, que ya está disponible de forma general en el blog. AWS

Contenido

- Preparación para asociar un grupo de destino de VPC Lattice a un grupo de escalado automático
- Asociación de un grupo de destino de VPC Lattice a su grupo de escalado automático
- Verificación del estado de asociación de su grupo de destino de VPC Lattice

Preparación para asociar un grupo de destino de VPC Lattice a un grupo de escalado automático

Antes de asociar un grupo de destino de VPC Lattice a un grupo de escalado automático, debe cumplir los siguientes requisitos previos:

- Debe haber creado ya una red de servicios, un servicio, un oyente y un grupo de destino de VPC Lattice. Para obtener más información, consulte los siguientes temas en la Guía del usuario de VPC Lattice:
  - Redes de servicios
  - Servicios
  - Oyentes
  - Grupos de destino
- El grupo objetivo debe estar en la misma Cuenta de AWS VPC y región que su grupo de Auto Scaling.
- Los grupos de destino deben especificar el tipo de destino `instance`. No puede especificar un tipo de destino `ip` cuando se utiliza un grupo de Auto Scaling.
- Usted debe tener suficientes permisos de IAM para asociar el grupo de destino al grupo de escalado automático. El siguiente ejemplo de política muestra los permisos mínimos necesarios para asociar y desasociar grupos de destino.

  JSON

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

```
            "Effect": "Allow",
            "Action": [
                "autoscaling:AttachTrafficSources",
                "autoscaling:DetachTrafficSources",
                "autoscaling:DescribeTrafficSources",
                "vpc-lattice:RegisterTargets",
                "vpc-lattice:DeregisterTargets"
            ],
            "Resource": "*"
        }
    ]
}
```

- Si la plantilla de lanzamiento de su grupo de escalado automático no contiene la configuración correcta para VPC Lattice, como un grupo de seguridad compatible, debe actualizar la plantilla de lanzamiento. Las instancias existentes no se actualizan con la nueva configuración cuando se modifica la plantilla de lanzamiento. Para actualizar las instancias existentes, puede comenzar una actualización de instancias para reemplazarlas. Para obtener más información, consulte Uso de una actualización de instancias para actualizar instancias en un grupo de escalado automático.
- Antes de habilitar las comprobaciones de estado de VPC Lattice en su grupo de escalado automático, puede configurar una comprobación de estado basada en la aplicación para comprobar que la aplicación responde según lo esperado. Para obtener más información, consulte Comprobaciones de estado de sus grupos de destino en la Guía del usuario de VPC Lattice.

Grupos de seguridad: reglas de entrada y salida

Los grupos de seguridad actúan como firewall para las instancias de EC2 asociadas al controlar el tráfico entrante y saliente en el ámbito de la instancia.

> **Note**
>
> La configuración de red es lo suficientemente compleja como para que le recomendemos ampliamente crear un nuevo grupo de seguridad para utilizarlo con VPC Lattice. También facilita su ayuda si necesita Soporte ponerse en contacto con ellos. Las siguientes secciones se basan en el supuesto de que se sigue esta recomendación.
>
> Para obtener más información sobre la creación de grupos de seguridad para VPC Lattice que pueda usar con su grupo de escalado automático, consulte Control del tráfico mediante

grupos de seguridad en la Guía del usuario de VPC Lattice. Para solucionar problemas de flujo de tráfico, consulte la Guía del usuario de VPC Lattice para obtener más información.

Para obtener información acerca de cómo crear un grupo de seguridad, consulte Crear un grupo de seguridad en la Guía del usuario de Amazon EC2 y utilice la siguiente tabla para determinar qué opciones seleccionar.

Opción	Valor
Name	Un nombre fácil de recordar.
Description (Descripción)	Una descripción que lo ayude a identificar el grupo de seguridad.
VPC	La misma VPC que el grupo de escalado automático.

Reglas de entrada

Cuando se crea un grupo de seguridad, este carece de reglas entrantes. No se permitirá el tráfico entrante que proceda de clientes de una red de servicios de VPC Lattice a su instancia hasta que no agregue reglas de entrada al grupo de seguridad.

Para permitir que los clientes de una red de servicios de VPC Lattice se conecten a las instancias de su grupo de escalado automático, el grupo de seguridad de su grupo de escalado automático debe estar configurado correctamente. En este caso, asígnele una regla de entrada para permitir el tráfico desde el nombre de la lista de AWS prefijos administrada para VPC Lattice, en lugar de desde una dirección IP específica. La lista de prefijos de VPC Lattice es un rango de direcciones IP que utiliza VPC Lattice en notación CIDR. Para obtener más información, consulte Trabajar con listas de AWS prefijos administradas en la Guía del usuario de Amazon VPC.

Para obtener información acerca de cómo agregar reglas a un grupo de seguridad, consulte Configurar las reglas del grupo de seguridad en la Guía del usuario de Amazon VPC y utilice la siguiente tabla para determinar qué opciones seleccionar.

Opción	Valor
Regla HTTP	Tipo: HTTP Fuente: com.amazonaws.*region*.vpc-lattice
Regla HTTPS	Tipo: HTTPS Fuente: com.amazonaws.*region*.vpc-lattice

El grupo de seguridad tiene estado: permite el tráfico desde los clientes de la red de servicios de VPC Lattice a las instancias del grupo de escalado automático y, a continuación, envía la respuesta de nuevo al cliente del que salió anteriormente.

Reglas de salida

De forma predeterminada, los grupos de seguridad incluyen una regla entrante que permite todo el tráfico saliente. Si lo desea, puede eliminar esta regla predeterminada y añadir una regla de salida para adaptarla a necesidades de seguridad específicas.

Limitaciones

- Se admiten grupos de instancias mixtas. Si asocia un grupo de destino de VPC Lattice a un grupo de escalado automático que tiene una política de instancias mixtas, el algoritmo del equilibrador de carga distribuye la carga de manera uniforme entre todos los recursos disponibles y supone que las instancias son lo suficientemente similares como para gestionar cargas iguales.

Asociación de un grupo de destino de VPC Lattice a su grupo de escalado automático

En este tema, se describe cómo asociar un grupo de destino de VPC Lattice a un grupo de escalado automático. También describe cómo activar las comprobaciones de estado de VPC Lattice para permitir que Amazon EC2 Auto Scaling sustituya las instancias que VPC Lattice informa que están en mal estado.

De forma predeterminada, Amazon EC2 Auto Scaling solo reemplaza las instancias en mal estado o inaccesibles en función de las comprobaciones de estado de Amazon EC2. Si activa las comprobaciones de estado de VPC Lattice, Amazon EC2 Auto Scaling puede reemplazar una instancia en ejecución si alguno de los grupos de destino de VPC Lattice que asocie al grupo de escalado automático informa que está en mal estado. Para obtener más información, consulte Comprobaciones de estado para instancias en un grupo de escalado automático.

> **Important**
>
> Antes de continuar, complete todos los requisitos previos de la sección anterior.

Asociación de un grupo de destino de VPC Lattice

Puede asociar uno o más grupos de destino a un grupo de escalado automático cuando crea o actualiza el grupo.

Console

Siga los pasos de esta sección para utilizar la consola a fin de hacer lo siguiente:

- Asociar un grupo de destino de VPC Lattice a un grupo de escalado automático
- Activación de las comprobaciones de estado de VPC Lattice

Asociación de un grupo de destino de VPC Lattice a un nuevo grupo de escalado automático

1. Abra la consola Amazon EC2 en https://console.aws.amazon.com/ec2/y seleccione Auto Scaling Groups en el panel de navegación.
2. En la barra de navegación de la parte superior de la pantalla, elija la Región de AWS en la que creó su grupo de destino.
3. Elija Create Auto Scaling group (Crear grupo de escalado automático).
4. En los pasos 1 y 2, elija las opciones que desee y continúe con el Paso 3: Configurar opciones avanzadas.
5. En Opciones de integración de VPC Lattice, elija Asociar al servicio de VPC Lattice.
6. En Elegir un grupo de destino de VPC Lattice, elija su grupo de destino.
7. (Opcional) En Comprobaciones de estado, Tipos de comprobaciones de estado adicionales, seleccione Activar las comprobaciones de estado de VPC Lattice.

8. (Opcional) En Período de gracia de comprobación de estado, ingrese el tiempo, en segundos. Este es la cantidad de tiempo que Amazon EC2 Auto Scaling debe esperar antes de comprobar el estado de una instancia una vez que pasa al estado `InService`. Para obtener más información, consulte Establezca el periodo de gracia de la comprobación de estado para un grupo de escalado automático.
9. Proceda a crear el grupo de escalado automático. Sus instancias se registrarán automáticamente en el grupo de destino de VPC Lattice una vez que haya creado el grupo de escalado automático.

Asociación de un grupo de destino de VPC Lattice a un grupo de escalado automático existente

Utilice el siguiente procedimiento para asociar un grupo de destino de un servicio a un grupo de escalado automático.

1. Abra la consola Amazon EC2 en https://console.aws.amazon.com/ec2/y seleccione Auto Scaling Groups en el panel de navegación.
2. Seleccione la casilla situada junto al grupo de escalado automático.

   Se abre un panel dividido en la parte inferior de la página.
3. En la pestaña Detalles, elija Opciones de integración de VPC Lattice, Editar.
4. En Opciones de integración de VPC Lattice, elija Asociar al servicio VPC de Lattice.
5. En Elegir un grupo de destino de VPC Lattice, elija su grupo de destino.
6. Elija Actualizar.

Cuando termine de asociar el grupo de destino, si lo desea, puede activar las comprobaciones de estado que lo utilizan.

Activación de las comprobaciones de estado de VPC Lattice

1. En la pestaña Details (Detalles), elija Health checks (Comprobaciones de estado), Edit (Editar).
2. En Comprobaciones de estado, Tipos de comprobaciones de estado adicionales, seleccione Activar las comprobaciones de estado de VPC Lattice.
3. En Período de gracia de comprobación de estado, ingrese el tiempo, en segundos. Este es la cantidad de tiempo que Amazon EC2 Auto Scaling debe esperar antes de comprobar el estado de una instancia una vez que pasa al estado `InService`. Para obtener más

información, consulte Establezca el periodo de gracia de la comprobación de estado para un grupo de escalado automático.

4. Elija Actualizar.

AWS CLI

Siga los pasos de esta sección para utilizarla AWS CLI para:

- Asociar un grupo de destino de VPC Lattice a un grupo de escalado automático
- Activación de las comprobaciones de estado de VPC Lattice

Asociación de un grupo de destino de VPC Lattice a un grupo de escalado automático

Utilice el siguiente comando create-auto-scaling-group para crear un grupo de escalado automático y asociar simultáneamente un grupo de destino de VPC Lattice especificando su nombre de recurso de Amazon (ARN).

Sustituya los valores de muestra de `--auto-scaling-group-name`, `--vpc-zone-identifier`, `--min-size` y `--max-size`. Para la opción `--launch-template`, sustituya *my-launch-template* y *1* por el nombre y la versión de la plantilla de lanzamiento que creó para las instancias registradas en un grupo de destino de VPC Lattice. Para la opción `--traffic-sources`, reemplace el ARN de muestra por el ARN del grupo de destino de VPC Lattice.

```
aws autoscaling create-auto-scaling-group --auto-scaling-group-name my-asg \
  --launch-template LaunchTemplateName=my-launch-template,Version='1' \
  --vpc-zone-identifier "subnet-5ea0c127,subnet-6194ea3b,subnet-c934b782" \
  --min-size 1 --max-size 5 \
  --traffic-sources "Identifier=arn:aws:vpc-lattice:region:account-id:targetgroup/
tg-0e2f2665eEXAMPLE"
```

Use el siguiente comando attach-traffic-sources para asociar un grupo de destino de VPC Lattice a un grupo de escalado automático una vez que ya esté creado.

```
aws autoscaling attach-traffic-sources --auto-scaling-group-name my-asg \
  --traffic-sources "Identifier=arn:aws:vpc-lattice:region:account-id:targetgroup/
tg-0e2f2665eEXAMPLE"
```

Activación de las comprobaciones de estado de VPC Lattice

Si ha configurado una comprobación de estado basada en aplicaciones para su grupo de destino de VPC Lattice, puede activar estas comprobaciones de estado. Utilice el comando create-auto-scaling-group o update-auto-scaling-group de la opción `--health-check-type` y un valor de **`VPC_LATTICE`**. Para especificar el período de gracia de las comprobaciones de estado realizadas por su grupo de escalado automático, incluya la opción `--health-check-grace-period` e indique su valor en segundos.

```
--health-check-type "VPC_LATTICE" --health-check-grace-period 60
```

Desasociación de un grupo de destino de VPC Lattice

Si ya no necesita utilizar VPC Lattice, utilice el siguiente procedimiento para desconectar el grupo de destino de su grupo de escalado automático.

Console

Siga los pasos de esta sección para utilizar la consola a fin de hacer lo siguiente:

- Separe un grupo de destino de VPC Lattice de un grupo de escalado automático
- Desactive las comprobaciones de estado de VPC Lattice

Desasociación de un grupo de destino de VPC Lattice de un grupo de escalado automático

1. Abra la consola Amazon EC2 en https://console.aws.amazon.com/ec2/y seleccione Auto Scaling Groups en el panel de navegación.
2. Seleccione la casilla de verificación situada junto a un grupo existente.

   Se abre un panel dividido en la parte inferior de la página.
3. En la pestaña Detalles, elija Opciones de integración de VPC Lattice, Editar.
4. En Opciones de integración de VPC Lattice, elija el icono de eliminación (X) situado junto al grupo de destino.
5. Elija Actualizar.

Cuando termine de desasociar el grupo de destino, podrá desactivar las comprobaciones de estado de VPC Lattice.

Desactivación de las comprobaciones de estado de VPC Lattice

1. En la pestaña Details (Detalles), elija Health checks (Comprobaciones de estado), Edit (Editar).
2. En Comprobaciones de estado, Tipos de comprobaciones de estado adicionales, anule la selección de Activar las comprobaciones de estado de VPC Lattice.
3. Elija Actualizar.

AWS CLI

Siga los pasos de esta sección para utilizarla AWS CLI para:

- Separe un grupo de destino de VPC Lattice de un grupo de escalado automático
- Desactive las comprobaciones de estado de VPC Lattice

Use el comando detach-traffic-sources para desasociar un grupo de destino de su grupo de escalado automático si ya no lo necesita.

```
aws autoscaling detach-traffic-sources --auto-scaling-group-name my-asg \
  --traffic-sources "Identifier=arn:aws:vpc-lattice:region:account-id:targetgroup/
tg-0e2f2665eEXAMPLE"
```

Para actualizar las comprobaciones de estado de un grupo de escalado automático que ya no utilice las comprobaciones de estado de VPC Lattice, utilice el comando update-auto-scaling-group. Incluya la opción `--health-check-type` y un valor de **`EC2`**.

```
aws autoscaling update-auto-scaling-group --auto-scaling-group-name my-asg \
  --health-check-type "EC2"
```

Verificación del estado de asociación de su grupo de destino de VPC Lattice

Después de asociar un grupo de destino de VPC Lattice a un grupo de escalado automático, este entra en el estado `Adding` a la vez que registra las instancias del grupo. Cuando se registran todas las instancias del grupo, entra en el estado `Added`. Cuando al menos una de las instancias registradas supera las comprobaciones de estado, pasa a tener el estado `InService`. Cuando

el grupo de destino se encuentra en el estado `InService`, Amazon EC2 Auto Scaling puede terminar y reemplazar las instancias notificadas como en mal estado. Si ninguna de las instancias registradas supera las comprobaciones de estado (debido, por ejemplo, a una comprobación de estado configurada incorrectamente), el grupo de destino no pasa al estado `InService`. Amazon EC2 Auto Scaling no termina y reemplaza las instancias.

Cuando desasocia un grupo de destino para un servicio, este pasa a tener el estado `Removing` mientras se cancela el registro de las instancias del grupo. Las instancias siguen ejecutándose una vez que se cancela el registro. De forma predeterminada, el drenaje de conexiones (retardo de cancelación del registro) está habilitado. Si el drenaje de conexiones está habilitado, VPC Lattice espera a que se completen las solicitudes en tránsito o a que termine el tiempo de espera máximo (lo que ocurra primero) antes de cancelar el registro de las instancias.

Puedes verificar el estado del archivo adjunto mediante AWS Command Line Interface (AWS CLI) o AWS los SDK. No puede verificar el estado de asociación desde la consola.

Para utilizar el AWS CLI para verificar el estado del archivo adjunto

El siguiente comando describe-traffic-sources devuelve el estado de asociación de todas las fuentes de tráfico del grupo de escalado automático especificado.

```
aws autoscaling describe-traffic-sources --auto-scaling-group-name my-asg
```

El ejemplo devuelve el ARN del grupo de destino de VPC Lattice que está asociado al grupo de escalado automático, junto con el estado de asociación del grupo de destino en el elemento `State`.

```
{
    "TrafficSources": [
        {
            "Identifier": "arn:aws:vpc-lattice:region:account-
id:targetgroup/tg-0e2f2665eEXAMPLE",
            "State": "InService",
            "Type": "vpc-lattice"
        }
    ]
}
```

Se usa EventBridge para gestionar eventos de Auto Scaling

Amazon EventBridge, anteriormente denominada CloudWatch Events, te ayuda a configurar reglas basadas en eventos que supervisan los recursos e inician acciones segmentadas que utilizan otros AWS servicios.

Los eventos de Amazon EC2 Auto Scaling se envían EventBridge prácticamente en tiempo real. Puede establecer EventBridge reglas que invoquen acciones y notificaciones programáticas en respuesta a una variedad de estos eventos. Por ejemplo, mientras las instancias se están iniciando o finalizando, puedes invocar una AWS Lambda función para realizar una tarea preconfigurada.

Los objetivos de EventBridge las reglas pueden incluir AWS Lambda funciones, temas de Amazon SNS, destinos de API, buses de eventos, entre otros Cuentas de AWS, y muchos más. Para obtener información sobre los objetivos admitidos, consulta EventBridge los objetivos de Amazon en la Guía del EventBridge usuario de Amazon.

Comience por crear EventBridge reglas con un ejemplo que utilice un tema de Amazon SNS y una EventBridge regla. A continuación, cuando un usuario inicia una actualización de instancias, Amazon SNS le notifica por correo electrónico cada vez que se alcanza un punto de control. Para obtener más información, consulte Cree EventBridge reglas (por ejemplo, actualice los eventos).

Contenido

- Referencia de evento de Amazon EC2 Auto Scaling
- Ejemplos de eventos y patrones de actualización de instancias
- Ejemplos de eventos y patrones de grupos en caliente
- Usa EventBridge las reglas de Amazon para automatizar las acciones

Referencia de evento de Amazon EC2 Auto Scaling

Con Amazon EventBridge, puedes crear reglas que coincidan con los eventos entrantes y enviarlos a los destinos para su procesamiento.

Contenido

- Eventos de acciones del ciclo de vida
- Eventos de escalado realizados correctamente
- Eventos de escalado no realizados correctamente

Eventos de acciones del ciclo de vida

Cuando agrega enlaces de ciclo de vida a su grupo de Auto Scaling, Amazon EC2 Auto Scaling envía eventos cuando una instancia pasa EventBridge a un estado de espera. Los eventos se producen en la medida de lo posible.

Event types (Tipos de eventos)

- Acción de escalar horizontalmente durante el ciclo de vida
- Acción de reducir horizontalmente durante el ciclo de vida

Acción de escalar horizontalmente durante el ciclo de vida

En el siguiente evento de ejemplo, se muestra que Amazon EC2 Auto Scaling ha movido una instancia al estado `Pending:Wait` debido a un enlace de ciclo de vida de lanzamiento.

```
{
  "version": "0",
  "id": "12345678-1234-1234-1234-123456789012",
  "detail-type": "EC2 Instance-launch Lifecycle Action",
  "source": "aws.autoscaling",
  "account": "123456789012",
  "time": "yyyy-mm-ddThh:mm:ssZ",
  "region": "us-west-2",
  "resources": [
    "auto-scaling-group-arn"
  ],
  "detail": {
    "LifecycleActionToken": "87654321-4321-4321-4321-210987654321",
    "AutoScalingGroupName": "my-asg",
    "LifecycleHookName": "my-lifecycle-hook",
    "EC2InstanceId": "i-1234567890abcdef0",
    "LifecycleTransition": "autoscaling:EC2_INSTANCE_LAUNCHING",
    "NotificationMetadata": "additional-info",
    "Origin": "EC2",
    "Destination": "AutoScalingGroup"
  }
}
```

Acción de reducir horizontalmente durante el ciclo de vida

En el siguiente evento de ejemplo, se muestra que Amazon EC2 Auto Scaling ha movido una instancia al estado `Terminating:Wait` debido a un enlace de ciclo de vida de finalización.

> **Important**
>
> Cuando un grupo de escalado automático devuelve las instancias a un grupo en caliente al reducir horizontalmente, el regreso de las instancias al grupo en caliente también puede generar eventos `EC2 Instance-terminate Lifecycle Action`. Los eventos que se entregan cuando una instancia pasa al estado de espera al reducir horizontalmente tienen `WarmPool` como valor para `Destination`. Para obtener más información, consulte Instance reuse policy.

```
{
  "version": "0",
  "id": "12345678-1234-1234-1234-123456789012",
  "detail-type": "EC2 Instance-terminate Lifecycle Action",
  "source": "aws.autoscaling",
  "account": "123456789012",
  "time": "yyyy-mm-ddThh:mm:ssZ",
  "region": "us-west-2",
  "resources": [
    "auto-scaling-group-arn"
  ],
  "detail": {
    "LifecycleActionToken":"87654321-4321-4321-4321-210987654321",
    "AutoScalingGroupName":"my-asg",
    "LifecycleHookName":"my-lifecycle-hook",
    "EC2InstanceId":"i-1234567890abcdef0",
    "LifecycleTransition":"autoscaling:EC2_INSTANCE_TERMINATING",
    "NotificationMetadata":"additional-info",
    "Origin": "AutoScalingGroup",
    "Destination": "EC2"
  }
}
```

Eventos de escalado realizados correctamente

En los siguientes ejemplos, se muestran los tipos de eventos necesarios para que los eventos de escalado se realicen correctamente. Los eventos se producen en la medida de lo posible.

Event types (Tipos de eventos)

- Evento de escalado horizontal realizado correctamente
- Evento de reducción horizontal realizado correctamente

Evento de escalado horizontal realizado correctamente

En el siguiente evento de ejemplo, se muestra que Amazon EC2 Auto Scaling lanzó una instancia correctamente.


```
{
  "version": "0",
  "id": "12345678-1234-1234-1234-123456789012",
  "detail-type": "EC2 Instance Launch Successful",
  "source": "aws.autoscaling",
  "account": "123456789012",
  "time": "yyyy-mm-ddThh:mm:ssZ",
  "region": "us-west-2",
  "resources": [
    "auto-scaling-group-arn",
    "instance-arn"
  ],
  "detail": {
    "StatusCode": "InProgress",
    "Description": "Launching a new EC2 instance: i-12345678",
    "AutoScalingGroupName": "my-asg",
    "ActivityId": "87654321-4321-4321-4321-210987654321",
    "Details": {
      "Availability Zone": "us-west-2b",
      "Subnet ID": "subnet-12345678"
    },
    "RequestId": "12345678-1234-1234-1234-123456789012",
    "StatusMessage": "",
    "EndTime": "yyyy-mm-ddThh:mm:ssZ",
    "EC2InstanceId": "i-1234567890abcdef0",
    "StartTime": "yyyy-mm-ddThh:mm:ssZ",
    "Cause": "description-text",
    "Origin": "EC2",
```

```
    "Destination": "AutoScalingGroup"
  }
}
```

Evento de reducción horizontal realizado correctamente

En el siguiente evento de ejemplo, se muestra que Amazon EC2 Auto Scaling finalizó una instancia correctamente.

> **Important**
>
> Cuando un grupo de escalado automático devuelve las instancias a un grupo en caliente al reducir horizontalmente, el regreso de las instancias al grupo en caliente también puede generar eventos `EC2 Instance Terminate Successful`. Los eventos que se entregan cuando una instancia regresa correctamente al grupo en caliente tienen `WarmPool` como valor para `Destination`. Para obtener más información, consulte Instance reuse policy.

```
{
  "version": "0",
  "id": "12345678-1234-1234-1234-123456789012",
  "detail-type": "EC2 Instance Terminate Successful",
  "source": "aws.autoscaling",
  "account": "123456789012",
  "time": "yyyy-mm-ddThh:mm:ssZ",
  "region": "us-west-2",
  "resources": [
    "auto-scaling-group-arn",
    "instance-arn"
  ],
  "detail": {
    "StatusCode": "InProgress",
    "Description": "Terminating EC2 instance: i-12345678",
    "AutoScalingGroupName": "my-asg",
    "ActivityId": "87654321-4321-4321-4321-210987654321",
    "Details": {
      "Availability Zone": "us-west-2b",
      "Subnet ID": "subnet-12345678"
    },
    "RequestId": "12345678-1234-1234-1234-123456789012",
    "StatusMessage": "",
```

```
    "EndTime": "yyyy-mm-ddThh:mm:ssZ",
    "EC2InstanceId": "i-1234567890abcdef0",
    "StartTime": "yyyy-mm-ddThh:mm:ssZ",
    "Cause": "description-text",
    "Origin": "AutoScalingGroup",
    "Destination": "EC2"
  }
}
```

Eventos de escalado no realizados correctamente

En los siguientes ejemplos, se muestran los tipos de eventos necesarios para que los eventos de escalado no se realicen correctamente. Los eventos se producen en la medida de lo posible.

Event types (Tipos de eventos)

- Evento de escalado horizontal no realizado correctamente
- Evento de reducción horizontal no realizado correctamente

Evento de escalado horizontal no realizado correctamente

En el siguiente evento de ejemplo, se muestra que Amazon EC2 Auto Scaling tuvo un error en el lanzamiento de una instancia.

```
{
  "version": "0",
  "id": "12345678-1234-1234-1234-123456789012",
  "detail-type": "EC2 Instance Launch Unsuccessful",
  "source": "aws.autoscaling",
  "account": "123456789012",
  "time": "yyyy-mm-ddThh:mm:ssZ",
  "region": "us-west-2",
  "resources": [
    "auto-scaling-group-arn",
    "instance-arn"
  ],
  "detail": {
    "StatusCode": "Failed",
    "AutoScalingGroupName": "my-asg",
    "ActivityId": "87654321-4321-4321-4321-210987654321",
    "Details": {
      "Availability Zone": "us-west-2b",
```

```
      "Subnet ID": "subnet-12345678"
    },
    "RequestId": "12345678-1234-1234-1234-123456789012",
    "StatusMessage": "message-text",
    "EndTime": "yyyy-mm-ddThh:mm:ssZ",
    "EC2InstanceId": "i-1234567890abcdef0",
    "StartTime": "yyyy-mm-ddThh:mm:ssZ",
    "Cause": "description-text",
    "Origin": "EC2",
    "Destination": "AutoScalingGroup"
  }
}
```

Evento de reducción horizontal no realizado correctamente

En el siguiente evento de ejemplo, se muestra que Amazon EC2 Auto Scaling tuvo un error en la finalización de una instancia.

> **Important**
>
> Cuando un grupo de escalado automático devuelve las instancias a un grupo en caliente al reducir horizontalmente, el hecho de no devolver las instancias al grupo en caliente también puede generar eventos `EC2 Instance Terminate Unsuccessful`. Los eventos que se entregan cuando una instancia no regresa correctamente al grupo en caliente tienen `WarmPool` como valor para `Destination`. Para obtener más información, consulte Instance reuse policy.

```
{
  "version": "0",
  "id": "12345678-1234-1234-1234-123456789012",
  "detail-type": "EC2 Instance Terminate Unsuccessful",
  "source": "aws.autoscaling",
  "account": "123456789012",
  "time": "yyyy-mm-ddThh:mm:ssZ",
  "region": "us-west-2",
  "resources": [
    "auto-scaling-group-arn",
    "instance-arn"
  ],
  "detail": {
```

```
    "StatusCode": "Failed",
    "AutoScalingGroupName": "my-asg",
    "ActivityId": "87654321-4321-4321-4321-210987654321",
    "Details": {
      "Availability Zone": "us-west-2b",
      "Subnet ID": "subnet-12345678"
    },
    "RequestId": "12345678-1234-1234-1234-123456789012",
    "StatusMessage": "message-text",
    "EndTime": "yyyy-mm-ddThh:mm:ssZ",
    "EC2InstanceId": "i-1234567890abcdef0",
    "StartTime": "yyyy-mm-ddThh:mm:ssZ",
    "Cause": "description-text",
    "Origin": "AutoScalingGroup",
    "Destination": "EC2"
  }
}
```

Ejemplos de eventos y patrones de actualización de instancias

Con Amazon EventBridge, puedes crear reglas que coincidan con los eventos entrantes y enviarlos a los destinos para su procesamiento.

Contenido

- Eventos de actualización de instancias
- La actualización de instancias reemplaza los eventos del ciclo de vida del volumen raíz

Eventos de actualización de instancias

Los siguientes ejemplos muestran eventos para la característica de actualización de instancias. Los eventos se producen en la medida de lo posible.

Event types (Tipos de eventos)

- Punto de comprobación alcanzado
- Actualización de instancia iniciada
- Actualización de instancia realizada satisfactoriamente
- Error en la actualización de instancia
- Cancelación de la actualización de instancia

- Inicio de la reversión de actualización de instancias
- Reversión de actualización de instancias realizada con éxito
- Error en la reversión de actualización de instancias

Punto de comprobación alcanzado

Cuando el número de instancias reemplazadas alcanza el umbral porcentual definido para el punto de comprobación, Amazon EC2 Auto Scaling emite el siguiente evento.

```
{
  "version": "0",
  "id": "12345678-1234-1234-1234-123456789012",
  "detail-type": "EC2 Auto Scaling Instance Refresh Checkpoint Reached",
  "source": "aws.autoscaling",
  "account": "123456789012",
  "time": "yyyy-mm-ddThh:mm:ssZ",
  "region": "us-west-2",
  "resources": [
    "auto-scaling-group-arn"
  ],
  "detail": {
    "InstanceRefreshId": "ab00cf8f-9126-4f3c-8010-dbb8cad6fb86",
    "AutoScalingGroupName": "my-asg",
    "CheckpointPercentage": "50",
    "CheckpointDelay": "300"
  }
}
```

Actualización de instancia iniciada

Cuando el estado de una actualización de una instancia cambia a `InProgress`, Amazon EC2 Auto Scaling emite el siguiente evento.

```
{
  "version": "0",
  "id": "12345678-1234-1234-1234-123456789012",
  "detail-type": "EC2 Auto Scaling Instance Refresh Started",
  "source": "aws.autoscaling",
  "account": "123456789012",
  "time": "yyyy-mm-ddThh:mm:ssZ",
  "region": "us-west-2",
```

```
  "resources": [
    "auto-scaling-group-arn"
  ],
  "detail": {
    "InstanceRefreshId": "c613620e-07e2-4ed2-a9e2-ef8258911ade",
    "AutoScalingGroupName": "my-asg"
  }
}
```

Actualización de instancia realizada satisfactoriamente

Cuando el estado de una actualización de una instancia cambia a `Successful`, Amazon EC2 Auto Scaling emite el siguiente evento.

```
{
  "version": "0",
  "id": "12345678-1234-1234-1234-123456789012",
  "detail-type": "EC2 Auto Scaling Instance Refresh Succeeded",
  "source": "aws.autoscaling",
  "account": "123456789012",
  "time": "yyyy-mm-ddThh:mm:ssZ",
  "region": "us-west-2",
  "resources": [
    "auto-scaling-group-arn"
  ],
  "detail": {
    "InstanceRefreshId": "c613620e-07e2-4ed2-a9e2-ef8258911ade",
    "AutoScalingGroupName": "my-asg"
  }
}
```

Error en la actualización de instancia

Cuando el estado de una actualización de una instancia cambia a `Failed`, Amazon EC2 Auto Scaling emite el siguiente evento.

```
{
  "version": "0",
  "id": "12345678-1234-1234-1234-123456789012",
  "detail-type": "EC2 Auto Scaling Instance Refresh Failed",
  "source": "aws.autoscaling",
  "account": "123456789012",
  "time": "yyyy-mm-ddThh:mm:ssZ",
```

```
  "region": "us-west-2",
  "resources": [
    "auto-scaling-group-arn"
  ],
  "detail": {
    "InstanceRefreshId": "c613620e-07e2-4ed2-a9e2-ef8258911ade",
    "AutoScalingGroupName": "my-asg"
  }
}
```

Cancelación de la actualización de instancia

Cuando el estado de una actualización de una instancia cambia a `Cancelled`, Amazon EC2 Auto Scaling emite el siguiente evento.

```
{
  "version": "0",
  "id": "12345678-1234-1234-1234-123456789012",
  "detail-type": "EC2 Auto Scaling Instance Refresh Cancelled",
  "source": "aws.autoscaling",
  "account": "123456789012",
  "time": "yyyy-mm-ddThh:mm:ssZ",
  "region": "us-west-2",
  "resources": [
    "auto-scaling-group-arn"
  ],
  "detail": {
    "InstanceRefreshId": "c613620e-07e2-4ed2-a9e2-ef8258911ade",
    "AutoScalingGroupName": "my-asg"
  }
}
```

Inicio de la reversión de actualización de instancias

Cuando el estado de una actualización de una instancia cambia a `RollbackInProgress`, Amazon EC2 Auto Scaling emite el siguiente evento.

```
{
  "version": "0",
  "id": "12345678-1234-1234-1234-123456789012",
  "detail-type": "EC2 Auto Scaling Instance Refresh Rollback Started",
  "source": "aws.autoscaling",
  "account": "123456789012",
```

```
  "time": "yyyy-mm-ddThh:mm:ssZ",
  "region": "us-west-2",
  "resources": [
    "auto-scaling-group-arn"
  ],
  "detail": {
    "InstanceRefreshId": "c613620e-07e2-4ed2-a9e2-ef8258911ade",
    "AutoScalingGroupName": "my-asg"
  }
}
```

Reversión de actualización de instancias realizada con éxito

Cuando el estado de una actualización de una instancia cambia a `RollbackSuccessful`, Amazon EC2 Auto Scaling emite el siguiente evento.

```
{
  "version": "0",
  "id": "12345678-1234-1234-1234-123456789012",
  "detail-type": "EC2 Auto Scaling Instance Refresh Rollback Succeeded",
  "source": "aws.autoscaling",
  "account": "123456789012",
  "time": "yyyy-mm-ddThh:mm:ssZ",
  "region": "us-west-2",
  "resources": [
    "auto-scaling-group-arn"
  ],
  "detail": {
    "InstanceRefreshId": "c613620e-07e2-4ed2-a9e2-ef8258911ade",
    "AutoScalingGroupName": "my-asg"
  }
}
```

Error en la reversión de actualización de instancias

Cuando el estado de una actualización de una instancia cambia a `Failed`, Amazon EC2 Auto Scaling emite el siguiente evento.

```
{
  "version": "0",
  "id": "12345678-1234-1234-1234-123456789012",
  "detail-type": "EC2 Auto Scaling Instance Refresh Rollback Failed",
  "source": "aws.autoscaling",
```

```
  "account": "123456789012",
  "time": "yyyy-mm-ddThh:mm:ssZ",
  "region": "us-west-2",
  "resources": [
    "auto-scaling-group-arn"
  ],
  "detail": {
    "InstanceRefreshId": "c613620e-07e2-4ed2-a9e2-ef8258911ade",
    "AutoScalingGroupName": "my-asg"
  }
}
```

La actualización de instancias reemplaza los eventos del ciclo de vida del volumen raíz

Cuando agrega enlaces de ciclo de vida a su grupo de Auto Scaling, Amazon EC2 Auto Scaling envía eventos cuando una instancia pasa EventBridge a un estado de espera. Para obtener más información, consulte Utilice enlaces de ciclo de vida para reemplazar el volumen raíz.

Los siguientes ejemplos muestran eventos en los que su grupo de Auto Scaling tiene instancias que se están sustituyendo por un volumen raíz durante una actualización de instancias. Los eventos se emiten en la medida de lo posible.

Ejemplos de evento

- Inicie la acción de reemplazar el volumen raíz durante el ciclo de vida
- Acción de finalizar el ciclo de vida del volumen raíz
- Ejemplo de patrones de eventos

Inicie la acción de reemplazar el volumen raíz durante el ciclo de vida

Cuando se selecciona una instancia para reemplazar el volumen raíz durante la actualización de la instancia y su estado cambia a`ReplacingRootVolume:Wait`.

```
{
  "version": "0",
  "id": "12345678-1234-1234-1234-123456789012",
  "detail-type": "EC2 Instance-terminate Lifecycle Action",
  "source": "aws.autoscaling",
  "account": "123456789012",
  "time": "2021-01-13T00:12:37.214Z",
  "region": "us-west-2",
```

```
  "resources": [
    "auto-scaling-group-arn"
  ],
  "detail": {
    "LifecycleActionToken": "71514b9d-6a40-4b26-8523-05e7eEXAMPLE",
    "AutoScalingGroupName": "my-asg",
    "LifecycleHookName": "my-terminate-lifecycle-hook",
    "EC2InstanceId": "i-1234567890abcdef0",
    "LifecycleTransition": "autoscaling:EC2_INSTANCE_TERMINATING",
    "NotificationMetadata": "additional-info",
    "Origin": "AutoScalingGroup",
    "Destination": "AutoScalingGroup",
    "Action": "ReplaceRootVolume"
  }
}
```

Acción de finalizar el ciclo de vida del volumen raíz

Cuando una instancia ha completado el reemplazo del volumen raíz y su estado cambia a`Pending:Wait`.

```
{
  "version": "0",
  "id": "12345678-1234-1234-1234-123456789012",
  "detail-type": "EC2 Instance-launch Lifecycle Action",
  "source": "aws.autoscaling",
  "account": "123456789012",
  "time": "2021-01-13T00:12:37.214Z",
  "region": "us-west-2",
  "resources": [
    "auto-scaling-group-arn"
  ],
  "detail": {
    "LifecycleActionToken": "71514b9d-6a40-4b26-8523-05e7eEXAMPLE",
    "AutoScalingGroupName": "my-asg",
    "LifecycleHookName": "my-launch-lifecycle-hook",
    "EC2InstanceId": "i-1234567890abcdef0",
    "LifecycleTransition": "autoscaling:EC2_INSTANCE_LAUNCHING",
    "NotificationMetadata": "additional-info",
    "Origin": "AutoScalingGroup",
    "Destination": "AutoScalingGroup",
    "Action": "ReplaceRootVolume"
  }
```

```
}
```

Ejemplo de patrones de eventos

La sección anterior proporciona ejemplos de eventos emitidos por Amazon EC2 Auto Scaling durante una actualización de instancias con una estrategia de reemplazo de volumen raíz. EventBridge los patrones de eventos tienen la misma estructura que los eventos con los que coinciden. El patrón cita los campos para los que se desea encontrar coincidencias y proporciona los valores que está buscando. Los siguientes campos del evento forman el patrón de evento definido en la regla para invocar una acción:

`"source": "aws.autoscaling"`

Identifica que el evento es de Amazon EC2 Auto Scaling.

`"detail-type": "EC2 Instance-launch Lifecycle Action"`

Identifica el tipo de evento.

`"Action": "ReplaceRootVolume"`

Identifica qué acción se está realizando en la instancia.

Usa el siguiente ejemplo de patrón de eventos para capturar todos los eventos asociados a las instancias que se preparan para reemplazar sus volúmenes raíz:

```
{
  "source": [ "aws.autoscaling" ],
  "detail-type": [ "EC2 Instance-terminate Lifecycle Action" ],
  "detail": {
      "Action": [ "ReplaceRootVolume" ]
   }
}
```

Utilice el siguiente ejemplo de patrón de eventos para capturar todos los eventos asociados a las instancias que vuelven a funcionar después de completar el reemplazo del volumen raíz:

```
{
  "source": [ "aws.autoscaling" ],
  "detail-type": [ "EC2 Instance-launch Lifecycle Action" ],
  "detail": {
      "Action": [ "ReplaceRootVolume" ]
```

```
    }
}
```

Ejemplos de eventos y patrones de grupos en caliente

Amazon EC2 Auto Scaling admite varios patrones predefinidos en Amazon EventBridge. Esto simplifica la creación de un patrón de eventos. Usted selecciona los valores de los campos en un formulario y EventBridge genera el patrón automáticamente. En este momento, Amazon EC2 Auto Scaling no admite patrones predefinidos para ningún evento emitido por un grupo de escalado automático con un grupo de calentamiento. Debe introducir el patrón como objeto JSON. Esta sección y el tema Crea EventBridge reglas para los eventos de piscina caliente muestran cómo utilizar un patrón de eventos para seleccionar eventos y enviarlos a los destinos.

Para crear EventBridge reglas que filtren los eventos relacionados con piscinas calientes a los que Amazon EC2 Auto Scaling EventBridge envía, incluya `Origin` los campos `Destination` y de `detail` la sección del evento.

Los valores de `Origin` y `Destination` pueden ser los siguientes:

`EC2` | `AutoScalingGroup` | `WarmPool`

Contenido

- Eventos de ejemplo
- Ejemplo de patrones de eventos

Eventos de ejemplo

Cuando agrega enlaces de ciclo de vida a su grupo de Auto Scaling, Amazon EC2 Auto Scaling envía eventos cuando una instancia pasa EventBridge a un estado de espera. Para obtener más información, consulte Uso de enlaces de ciclo de vida con un grupo de calentamiento en un grupo de escalado automático.

Esta sección incluye ejemplos de estos eventos cuando su grupo de escalado automático tiene un grupo en caliente. Los eventos se emiten en la medida de lo posible.

> **Note**
>
> Para ver los eventos a los que Amazon EC2 Auto Scaling envía cuando EventBridge el escalado se realiza correctamente, Eventos de escalado realizados correctamente consulte.

Para ver los eventos en los que el escalado no se realiza correctamente, consulte Eventos de escalado no realizados correctamente.

Ejemplos de evento

- Acción de escalar horizontalmente durante el ciclo de vida
- Acción de reducir horizontalmente durante el ciclo de vida

Acción de escalar horizontalmente durante el ciclo de vida

Los eventos que se entregan cuando una instancia pasa al estado de espera debido a eventos de escalado horizontal tienen `EC2 Instance-launch Lifecycle Action` como valor para `detail-type`. En el objeto `detail`, los valores de los atributos `Origin` y `Destination` muestran el origen y el destino de la instancia.

En este ejemplo de evento de escalado horizontal, se lanza una nueva instancia y su estado cambia a `Warmed:Pending:Wait` porque se agrega al grupo en caliente. Para obtener más información, consulte Transiciones de estado del ciclo de vida para las instancias de un grupo de calentamiento.

```
{
  "version": "0",
  "id": "12345678-1234-1234-1234-123456789012",
  "detail-type": "EC2 Instance-launch Lifecycle Action",
  "source": "aws.autoscaling",
  "account": "123456789012",
  "time": "2021-01-13T00:12:37.214Z",
  "region": "us-west-2",
  "resources": [
    "auto-scaling-group-arn"
  ],
  "detail": {
    "LifecycleActionToken": "71514b9d-6a40-4b26-8523-05e7eEXAMPLE",
    "AutoScalingGroupName": "my-asg",
    "LifecycleHookName": "my-launch-lifecycle-hook",
    "EC2InstanceId": "i-1234567890abcdef0",
    "LifecycleTransition": "autoscaling:EC2_INSTANCE_LAUNCHING",
    "NotificationMetadata": "additional-info",
    "Origin": "EC2",
    "Destination": "WarmPool"
  }
```

```
}
```

En este ejemplo de escalado horizontal, el estado de la instancia cambia a `Pending:Wait` porque se la agrega al grupo de escalado automático del grupo en caliente. Para obtener más información, consulte Transiciones de estado del ciclo de vida para las instancias de un grupo de calentamiento.

```
{
  "version": "0",
  "id": "12345678-1234-1234-1234-123456789012",
  "detail-type": "EC2 Instance-launch Lifecycle Action",
  "source": "aws.autoscaling",
  "account": "123456789012",
  "time": "2021-01-19T00:35:52.359Z",
  "region": "us-west-2",
  "resources": [
    "auto-scaling-group-arn"
  ],
  "detail": {
    "LifecycleActionToken": "19cc4d4a-e450-4d1c-b448-0de67EXAMPLE",
    "AutoScalingGroupName": "my-asg",
    "LifecycleHookName": "my-launch-lifecycle-hook",
    "EC2InstanceId": "i-1234567890abcdef0",
    "LifecycleTransition": "autoscaling:EC2_INSTANCE_LAUNCHING",
    "NotificationMetadata": "additional-info",
    "Origin": "WarmPool",
    "Destination": "AutoScalingGroup"
  }
}
```

Acción de reducir horizontalmente durante el ciclo de vida

Los eventos que se entregan cuando una instancia pasa al estado de espera debido a eventos de reducción horizontal tienen `EC2 Instance-terminate Lifecycle Action` como valor para `detail-type`. En el objeto `detail`, los valores de los atributos `Origin` y `Destination` muestran el origen y el destino de la instancia.

En este evento de ejemplo de reducción horizontal, el estado de una instancia cambia a `Warmed:Pending:Wait` porque se devuelve al grupo en caliente. Para obtener más información, consulte Transiciones de estado del ciclo de vida para las instancias de un grupo de calentamiento.

```
{
```

```
  "version": "0",
  "id": "12345678-1234-1234-1234-123456789012",
  "detail-type": "EC2 Instance-terminate Lifecycle Action",
  "source": "aws.autoscaling",
  "account": "123456789012",
  "time": "2022-03-28T00:12:37.214Z",
  "region": "us-west-2",
  "resources": [
    "auto-scaling-group-arn"
  ],
  "detail": {
    "LifecycleActionToken": "42694b3d-4b70-6a62-8523-09a1eEXAMPLE",
    "AutoScalingGroupName": "my-asg",
    "LifecycleHookName": "my-termination-lifecycle-hook",
    "EC2InstanceId": "i-1234567890abcdef0",
    "LifecycleTransition": "autoscaling:EC2_INSTANCE_TERMINATING",
    "NotificationMetadata": "additional-info",
    "Origin": "AutoScalingGroup",
    "Destination": "WarmPool"
  }
}
```

Ejemplo de patrones de eventos

En la sección anterior se dan eventos de ejemplo emitidos por Amazon EC2 Auto Scaling.

EventBridge los patrones de eventos tienen la misma estructura que los eventos con los que coinciden. El patrón cita los campos para los que se desea encontrar coincidencias y proporciona los valores que está buscando.

Los siguientes campos del evento forman el patrón de evento definido en la regla para invocar una acción:

`"source": "aws.autoscaling"`

Identifica que el evento es de Amazon EC2 Auto Scaling.

`"detail-type": "EC2 Instance-launch Lifecycle Action"`

Identifica el tipo de evento.

`"Origin": "EC2"`

Identifica de dónde proviene la instancia.

`"Destination": "WarmPool"`

Identifica a dónde va la instancia.

Utilice el siguiente patrón de eventos de ejemplo para capturar todos los eventos de `EC2 Instance-launch Lifecycle Action` asociados a las instancias que entran en el grupo en caliente.

```
{
  "source": [ "aws.autoscaling" ],
  "detail-type": [ "EC2 Instance-launch Lifecycle Action" ],
  "detail": {
      "Origin": [ "EC2" ],
      "Destination": [ "WarmPool" ]
   }
}
```

Utilice el siguiente patrón de eventos de ejemplo para capturar todos los eventos de `EC2 Instance-launch Lifecycle Action` asociados a las instancias que salen del grupo en caliente debido a un escalado horizontal.

```
{
  "source": [ "aws.autoscaling" ],
  "detail-type": [ "EC2 Instance-launch Lifecycle Action" ],
  "detail": {
      "Origin": [ "WarmPool" ],
      "Destination": [ "AutoScalingGroup" ]
   }
}
```

Utilice el siguiente patrón de eventos de ejemplo para capturar todos los eventos de `EC2 Instance-launch Lifecycle Action` asociados a las instancias que se lanzan directamente al grupo de escalado automático.

```
{
  "source": [ "aws.autoscaling" ],
  "detail-type": [ "EC2 Instance-launch Lifecycle Action" ],
  "detail": {
      "Origin": [ "EC2" ],
      "Destination": [ "AutoScalingGroup" ]
```

```
    }
}
```

Utilice el siguiente patrón de eventos de ejemplo para capturar todos los eventos de `EC2 Instance-terminate Lifecycle Action` asociados a las instancias que vuelven al grupo en caliente debido a una reducción horizontal.

```
{
  "source": [ "aws.autoscaling" ],
  "detail-type": [ "EC2 Instance-terminate Lifecycle Action" ],
  "detail": {
      "Origin": [ "AutoScalingGroup" ],
      "Destination": [ "WarmPool" ]
   }
}
```

Utilice el siguiente patrón de eventos de ejemplo para capturar todos los eventos asociados a `EC2 Instance-launch Lifecycle Action`, independientemente del origen o el destino.

```
{
  "source": [ "aws.autoscaling" ],
  "detail-type": [ "EC2 Instance-launch Lifecycle Action" ]
}
```

Usa EventBridge las reglas de Amazon para automatizar las acciones

Cuando Amazon EC2 Auto Scaling emite un evento, se envía una notificación de evento a EventBridge Amazon como un archivo JSON. Puede escribir una EventBridge regla para automatizar las acciones que se deben realizar cuando un patrón de eventos coincide con la regla. Si EventBridge detecta un patrón de eventos que coincide con un patrón definido en una regla, EventBridge invoca el objetivo (o los objetivos) especificados en la regla.

Puede utilizar los procedimientos de ejemplo de esta sección como punto de partida.

La siguiente documentación también puede serle de utilidad.

- Para realizar acciones personalizadas en las instancias a medida que se están lanzando o antes de que finalicen mediante una función Lambda, consulte Tutorial: Configuración de un enlace de ciclo de vida que invoca una función Lambda.

- Para invocar una función de Lambda en las llamadas a la API registradas CloudTrail, consulte el Tutorial: AWS Registrar llamadas a la API EventBridge mediante el uso de la Guía del usuario de EventBridge Amazon.
- Para obtener más información sobre cómo crear reglas de eventos, consulta Cómo crear EventBridge reglas de Amazon que reaccionen a los eventos en la Guía del EventBridge usuario de Amazon.

Temas

- Cree EventBridge reglas (por ejemplo, actualice los eventos)
- Crea EventBridge reglas para los eventos de piscina caliente

Cree EventBridge reglas (por ejemplo, actualice los eventos)

En el siguiente ejemplo, se crea una EventBridge regla para enviar una notificación por correo electrónico. Esto lo hace cada vez que el grupo de escalado automático emite un evento cuando se alcanza un punto de control durante la actualización de instancias. Se incluye el procedimiento para configurar notificaciones por correo electrónico con Amazon SNS. Para utilizar Amazon SNS para enviar notificaciones por correo electrónico, primero debe crear un tema y, a continuación, suscribir sus direcciones de correo electrónico al tema.

Para obtener más información sobre la función de actualización de instancias, consulte Uso de una actualización de instancias para actualizar instancias en un grupo de escalado automático.

Crear un tema de Amazon SNS

Un tema de SNS es un punto de acceso lógico, un canal de comunicación que su grupo de Auto Scaling utiliza para enviar las notificaciones. Los temas se crean especificando un nombre para el tema.

Los nombres de los temas deben cumplir con los siguientes requisitos:

- Tener de 1 a 256 caracteres
- Deben contener letras ASCII en mayúsculas y minúsculas, números, guiones bajos o guiones.

Para obtener instrucciones, consulte el tema Creación de un tema de Amazon SNS en la Guía para desarrolladores de Amazon Simple Notification Service.

Suscripción al tema de Amazon SNS

Para recibir las notificaciones que su grupo de escalado automático envía al tema, debe suscribir un punto de conexión al tema. En este procedimiento, en Punto de conexión, especifique la dirección de correo electrónico donde desea recibir las notificaciones de Amazon EC2 Auto Scaling.

Para obtener más información, consulte el tema Suscripción a un tema de Amazon SNS en la Guía para desarrolladores de Amazon Simple Notification Service.

Confirmación de la suscripción a Amazon SNS

Amazon SNS envía un correo electrónico de confirmación a la dirección de correo electrónico que ha especificado en el paso anterior.

Asegúrese de abrir el correo electrónico desde AWS Notificaciones y de elegir el enlace para confirmar la suscripción antes de continuar con el siguiente paso.

Recibirás un mensaje de confirmación de AWS. Amazon SNS estará ahora configurado para recibir notificaciones y enviar la notificación como un email a la dirección especificada.

Enrutamiento de los eventos al tema de Amazon SNS

Cree una regla que coincida con los eventos seleccionados y los dirija al tema de Amazon SNS para notificar a las direcciones de correo electrónico suscritas.

Para crear una regla que envíe notificaciones al tema de Amazon SNS

1. Abre la EventBridge consola de Amazon en https://console.aws.amazon.com/events/.
2. En el panel de navegación, seleccione Reglas.
3. Seleccione Creación de regla.
4. En Definir detalle de la regla, haga lo siguiente:

    a. Ingrese un Nombre para la regla y, opcionalmente, una descripción.

        Una regla no puede tener el mismo nombre que otra regla de la misma región y del mismo bus de eventos.

    b. En Bus de eventos, elija Predeterminado. Cuando un AWS servicio de tu cuenta genera un evento, siempre va al bus de eventos predeterminado de tu cuenta.

    c. En Tipo de regla, elija Regla con un patrón de evento.

d. Elija Siguiente.

5. En Crear patrón de evento, realice una de las siguientes acciones:

   a. En Origen del evento, selecciona AWS eventos o eventos EventBridge asociados.

   b. En Event pattern (Patrón de eventos), realice una de las siguientes acciones:

      i. En Origen del evento, elija Servicios de AWS.

      ii. En Servicio de AWS, elija Auto Scaling.

      iii. En Event type (Tipo de evento), elija Instance Refresh (Actualización de instancias).

      iv. De forma predeterminada, la regla coincide con cualquier evento de actualización de instancias. Para crear una regla que le notifique cada vez que se alcance un punto de control durante la actualización de instancias, elija Specific instance event(s) (Eventos de instancia específicos) y seleccione EC2 Auto Scaling Instance Refresh Checkpoint Reached (Se ha alcanzado un punto de control de actualización de instancias de EC2 Auto Scaling).

      v. De forma predeterminada, la regla coincide con cualquier grupo de Auto Scaling en la región. Para que la regla coincida con un grupo de Auto Scaling específico, elija Specific group name(s) (Nombres de grupos específicos) y, a continuación, seleccione uno o varios grupos de Auto Scaling.

      vi. Elija Siguiente.

6. En Seleccionar destino, realice una de las siguientes acciones:

   a. Para Target types (Tipos de destino), elija Servicio de AWS.

   b. Para Select a target (Seleccione un destino), elija SNS topic (Tema de SNS).

   c. En Topic (Tema), elija su tema de Amazon SNS.

   d. (Opcional) En Configuración adicional, puede configurar opciones adicionales. Para obtener más información, consulta Cómo crear EventBridge reglas de Amazon que reaccionen a los eventos en la Guía del EventBridge usuario de Amazon.

   e. Elija Siguiente.

7. (Opcional) En Etiquetas, puede asignar una o varias etiquetas a la regla y, a continuación, elija Siguiente.

8. En Review and create (Revisar y crear), revise los detalles de la regla y modifíquelos según sea necesario. A continuación, elija Create rule (Crear regla).

Crea EventBridge reglas para los eventos de piscina caliente

En el siguiente ejemplo, se crea una EventBridge regla para invocar acciones programáticas. Esto lo hace cada vez que el grupo de escalado automático emite un evento cuando se agrega una nueva instancia al grupo de calentamiento.

Antes de crear la regla, cree la AWS Lambda función que desee que utilice la regla como destino. Debe especificar esta función como destino. El siguiente procedimiento proporciona solo los pasos para crear la EventBridge regla que actúa cuando entran nuevas instancias en la piscina caliente. Para obtener una guía introductoria que le muestre cómo crear una función de Lambda simple para invocar cuando un evento entrante coincide con una regla, consulte Tutorial: Configuración de un enlace de ciclo de vida que invoca una función Lambda.

Para obtener más información sobre cómo crear y trabajar con grupos de calentamiento, consulte Reducción de la latencia de las aplicaciones con tiempos de arranque prolongados mediante grupos de calentamiento.

Para crear una regla de evento que invoque una función de Lambda

1. Abre la EventBridge consola de Amazon en https://console.aws.amazon.com/events/.
2. En el panel de navegación, seleccione Reglas.
3. Seleccione Creación de regla.
4. En Definir detalle de la regla, haga lo siguiente:

    a. Ingrese un Nombre para la regla y, opcionalmente, una descripción.

    Una regla no puede tener el mismo nombre que otra regla de la misma región y del mismo bus de eventos.

    b. En Bus de eventos, elija Predeterminado. Cuando un Servicio de AWS elemento de tu cuenta genera un evento, siempre va al bus de eventos predeterminado de tu cuenta.

    c. En Tipo de regla, elija Regla con un patrón de evento.

    d. Elija Siguiente.
5. En Crear patrón de evento, realice una de las siguientes acciones:

    a. En Origen del evento, selecciona AWS eventos o eventos EventBridge asociados.

    b. En Patrón de eventos, elija Patrón personalizado (editor JSON) y pegue el siguiente patrón en el cuadro Patrón de eventos, sustituyendo el texto por el nombre de su grupo de Auto Scaling. *italics*

```
{
  "source": [ "aws.autoscaling" ],
  "detail-type": [ "EC2 Instance-launch Lifecycle Action" ],
  "detail": {
      "AutoScalingGroupName": [ "my-asg" ],
      "Origin": [ "EC2" ],
      "Destination": [ "WarmPool" ]
   }
}
```

Para crear una regla que coincida con otros eventos, modifique el patrón de eventos. Para obtener más información, consulte Ejemplo de patrones de eventos.

c. Elija Siguiente.

6. En Seleccionar destino, realice una de las siguientes acciones:

   a. Para Target types (Tipos de destino), elija Servicio de AWS.

   b. En Target (Destino), elija Lambda function (Función de Lambda).

   c. Para Function (Función), elija la función a la que quiera enviar los eventos.

   d. (Opcional) En Configure version/alias (Configurar la versión o el alias), ingrese la configuración de versión y alias de la función de Lambda de destino.

   e. (Opcional) En Additional settings (Configuración adicional), ingrese cualquier configuración adicional según sea apropiado para su aplicación. Para obtener más información, consulta Cómo crear EventBridge reglas de Amazon que reaccionen a los eventos en la Guía del EventBridge usuario de Amazon.

   f. Elija Siguiente.

7. (Opcional) En Etiquetas, puede asignar una o varias etiquetas a la regla y, a continuación, elija Siguiente.

8. En Review and create (Revisar y crear), revise los detalles de la regla y modifíquelos según sea necesario. A continuación, elija Create rule (Crear regla).

Proporcionar conectividad de red para sus instancias de Auto Scaling mediante Amazon VPC

Amazon Virtual Private Cloud (Amazon VPC) es un servicio que le permite lanzar AWS recursos como grupos de Auto Scaling en una red virtual aislada de forma lógica que usted defina.

Una subred de Amazon VPC es una subdivisión dentro de una zona de disponibilidad definida por un segmento del intervalo de direcciones IP de la VPC. Mediante el uso de subredes, puede agrupar sus instancias en función de sus necesidades operativas y de seguridad. Una subred reside en su totalidad dentro de la zona de disponibilidad en la que se creó. Las instancias de Auto Scaling se lanzan dentro de las subredes.

Para permitir la comunicación entre Internet y las instancias de las subredes, debe crear una gateway de Internet y asociarla a la VPC. Una gateway de Internet permite que los recursos incluidos en las subredes se conecten a Internet a través del límite de la red de Amazon EC2. Si el tráfico de una subred se direcciona a un puerto de enlace a Internet, la subred recibe el nombre de subred pública. Si el tráfico de una subred no se dirige a una gateway de Internet, la subred recibe el nombre de subred privada. Utilice una subred pública para los recursos que deben conectarse a Internet y una subred privada para los recursos que no necesitan conectarse a Internet. Para obtener más información acerca de cómo proporcionar acceso a Internet a instancias en una VPC, consulte Acceder a Internet en la Guía del usuario de Amazon VPC.

Contenido

- VPC predeterminada
- VPC no predeterminada
- Consideraciones a la hora de elegir subredes de VPC
- Direcciones IP en una VPC
- Interfaces de red en una VPC
- Tenencia de ubicación de instancias
- AWS Outposts
- Más recursos para obtener más información sobre VPCs

VPC predeterminada

Si creó su grupo Cuenta de AWS después del 4 de diciembre de 2013 o si va a crear su grupo de Auto Scaling en uno nuevo Región de AWS, crearemos una VPC predeterminada para usted. La VPC predeterminada incluye una subred predeterminada en cada zona de disponibilidad. Si dispone de una VPC predeterminada, su grupo de escalado automático se crea en la VPC predeterminada, de forma predeterminada.

Puedes verla VPCs en la VPCs página Tu de la consola de Amazon VPC.

Para obtener más información sobre la VPC predeterminada, consulte Predeterminado VPCs en la Guía del usuario de Amazon VPC.

VPC no predeterminada

Puede optar VPCs por crear más. Para ello, vaya a la página del panel de control de VPC Consola de administración de AWS y seleccione Crear VPC.

Para obtener más información, consulte la Guía del usuario de Amazon VPC.

> **Note**
>
> Un VPC abarca todas las zonas de disponibilidad en su Región de AWS. Cuando agregue subredes a la VPC, elija varias zonas de disponibilidad para asegurarse de que las aplicaciones alojadas en esas subredes estén altamente disponibles. Una zona de disponibilidad consiste en uno o varios centros de datos discretos con alimentación, redes y conectividad redundantes en una Región de AWS. Las zonas de disponibilidad permiten que las aplicaciones de producción sean altamente disponibles, tolerantes a errores y tengan escalabilidad.

Consideraciones a la hora de elegir subredes de VPC

Tenga en cuenta las siguientes consideraciones al elegir subredes VPC para su grupo de escalado automático:

- Si va a adjuntar un balanceador de carga de Elastic Load Balancing al grupo de escalado automático, las instancias se pueden iniciar en subredes públicas o privadas. Sin embargo, el equilibrador de carga se debe crear en subredes públicas para admitir la resolución de DNS.

- Si accede a las instancias de Auto Scaling directamente a través de SSH, las instancias solo se pueden iniciar en subredes públicas.
- Si accede a instancias de Auto Scaling sin entrada mediante el Administrador de AWS Systems Manager sesiones, las instancias se pueden lanzar en subredes públicas o privadas.
- Si utiliza subredes privadas, puede permitir que las instancias de Auto Scaling accedan a Internet mediante un gateway NAT público.
- De forma predeterminada, las subredes predeterminadas de una VPC predeterminada son subredes públicas.

Direcciones IP en una VPC

Cuando lanza instancias Auto Scaling en una VPC, las instancias reciben automáticamente una dirección IP privada del intervalo de CIDR de la subred en la que se lanza la instancia. Esto permite que las instancias se comuniquen con otras instancias en la VPC.

Puede configurar una plantilla de lanzamiento o una configuración de lanzamiento para asignar IPv4 direcciones públicas a sus instancias. La asignación de direcciones IP públicas a sus instancias les permite comunicarse con Internet u otros AWS servicios.

Cuando lanzas instancias a una subred que está configurada para asignar IPv6 direcciones automáticamente, reciben ambas IPv4 direcciones. IPv6 De lo contrario, solo reciben IPv4 direcciones. Para obtener más información, consulte IPv6las direcciones en la Guía del usuario de Amazon EC2.

Para obtener información sobre cómo especificar intervalos de CIDR para su VPC o subred, consulte Guía del usuario de Amazon VPC.

Amazon EC2 Auto Scaling puede asignar automáticamente direcciones IP privadas adicionales en el lanzamiento de las instancias cuando se utiliza una plantilla de lanzamiento que especifica interfaces de red adicionales. A cada interfaz de red se le asigna una única dirección IP privada del intervalo de CIDR de la subred en la que se lanza la instancia. En este caso, el sistema ya no puede asignar automáticamente una IPv4 dirección pública a la interfaz de red principal. No podrá conectarse a sus instancias a través de una IPv4 dirección pública a menos que asocie las direcciones IP elásticas disponibles a las instancias de Auto Scaling.

Interfaces de red en una VPC

Cada instancia de su VPC tiene una interfaz de red predeterminada (la interfaz de red principal). No se puede desconectar una interfaz de red principal de una instancia. Puede crear y adjuntar una interfaz de red adicional a cualquier instancia de su VPC. El número total de interfaces de red que puede adjuntar varía en función del tipo de instancia.

Cuando lanza una instancia utilizando una plantilla de lanzamiento, puede especificar interfaces de red adicionales. Sin embargo, al iniciar una instancia de Auto Scaling con varias interfaces de red, cada interfaz se crea automáticamente en la misma subred que la instancia. Esto se debe a que Amazon EC2 Auto Scaling ignora las subredes definidas en la plantilla de lanzamiento en favor de lo especificado en el grupo de escalado automático. Para obtener más información, consulte Creación de una plantilla de lanzamiento para un grupo de escalado automático.

Si crea o asocia dos o más interfaces de red de la misma subred a una instancia, pueden producirse problemas de red, como el direccionamiento asimétrico, especialmente en instancias que utilizan una variante que no es Amazon Linux. Si necesita este tipo de configuración, debe configurar la interfaz de red secundaria dentro del sistema operativo. Para ver un ejemplo, consulta ¿Cómo puedo hacer que mi interfaz de red secundaria funcione en mi instancia EC2 de Ubuntu? en el Centro de AWS conocimiento.

Tenencia de ubicación de instancias

De forma predeterminada, todas las instancias de la VPC se ejecutan como instancias de tenencia compartida. Amazon EC2 Auto Scaling también admite instancias dedicadas y hosts dedicados. Para obtener más información, consulte Creación de una plantilla de lanzamiento mediante la configuración avanzada.

AWS Outposts

AWS Outposts extiende una VPC de Amazon de una AWS región a un puesto avanzado con los componentes de VPC a los que se puede acceder en la región, incluidas las puertas de enlace de Internet, las puertas de enlace privadas virtuales, las pasarelas de tránsito de Amazon VPC y los puntos de enlace de VPC. Un Outpost está destinado a una zona de disponibilidad de la región y es una extensión de esa zona de disponibilidad que puede utilizar para obtener resiliencia.

Para obtener más información, consulte la Guía del usuario de AWS Outposts.

Para ver un ejemplo de cómo implementar un grupo de escalado automático que proporciona tráfico desde un equilibrador de carga de aplicación dentro de una instancia de Outposts, consulte

la siguiente publicación de blog Configuración de un Equilibrador de carga de aplicación en AWS Outposts.

Más recursos para obtener más información sobre VPCs

Utilice los siguientes temas para obtener más información sobre las subredes VPCs y las subredes.

- Subredes privadas en una VPC
  - Ejemplo: una VPC con servidores en subredes privadas y NAT
  - Gateways NAT
- Subredes públicas en una VPC
  - Ejemplo: VPC para un entorno de prueba
  - Ejemplo: una VPC para servidores web y de bases de datos
- Subredes del Application Load Balancer
  - Subredes del equilibrador de carga
- Información de VPC general
  - Guía del usuario de Amazon VPC
  - Conéctese VPCs mediante el emparejamiento de VPC
  - Interfaces de red elásticas
  - Uso de puntos de conexión de VPC para conectividad privada

Seguridad en Amazon EC2 Auto Scaling

La seguridad en la nube AWS es la máxima prioridad. Como AWS cliente, usted se beneficia de una arquitectura de centro de datos y red diseñada para cumplir con los requisitos de las organizaciones más sensibles a la seguridad.

La seguridad es una responsabilidad compartida entre usted AWS y usted. El modelo de responsabilidad compartida la describe como seguridad de la nube y seguridad en la nube:

- Seguridad de la nube: AWS es responsable de proteger la infraestructura que ejecuta AWS los servicios en la AWS nube. AWS también le proporciona servicios que puede utilizar de forma segura. Los auditores externos prueban y verifican periódicamente la eficacia de nuestra seguridad como parte de los AWS programas de de . Para obtener más información sobre los programas de conformidad que se aplican a Amazon EC2 Auto Scaling, AWS consulte los servicios incluidos en el ámbito por programa de conformidad y .
- Seguridad en la nube: su responsabilidad viene determinada por el AWS servicio que utilice. También es responsable de otros factores, incluida la confidencialidad de los datos, los requisitos de la empresa y la legislación y los reglamentos vigentes.

Esta documentación le ayuda a comprender cómo aplicar el modelo de responsabilidad compartida al utilizar Amazon EC2 Auto Scaling. En los siguientes temas, se mostrará cómo configurar Amazon EC2 Auto Scaling para satisfacer sus objetivos de seguridad y conformidad. También aprenderá a utilizar otros AWS servicios que le ayudan a supervisar y proteger sus recursos de Amazon EC2 Auto Scaling.

Temas

- Seguridad de la infraestructura de Amazon EC2 Auto Scaling
- Resiliencia en Amazon EC2 Auto Scaling
- Protección de los datos en Amazon EC2 Auto Scaling
- Identity and Access Management para Amazon EC2 Auto Scaling
- Validación de la conformidad en Amazon EC2 Auto Scaling
- Amazon EC2 Auto Scaling y puntos de conexión de VPC de tipo interfaz

Seguridad de la infraestructura de Amazon EC2 Auto Scaling

Como servicio gestionado, Amazon EC2 Auto Scaling está protegido por la seguridad de la red AWS global. Para obtener información sobre los servicios AWS de seguridad y cómo se AWS protege la infraestructura, consulte Seguridad AWS en la nube. Para diseñar su AWS entorno utilizando las mejores prácticas de seguridad de la infraestructura, consulte Protección de infraestructuras en un marco de buena AWS arquitectura basado en el pilar de la seguridad.

Utilice las llamadas a la API AWS publicadas para acceder a Amazon EC2 Auto Scaling a través de la red. Los clientes deben admitir lo siguiente:

- Seguridad de la capa de transporte (TLS). Exigimos TLS 1.2 y recomendamos TLS 1.3.
- Conjuntos de cifrado con confidencialidad directa total (PFS) como DHE (Ephemeral Diffie-Hellman) o ECDHE (Elliptic Curve Ephemeral Diffie-Hellman). La mayoría de los sistemas modernos como Java 7 y posteriores son compatibles con estos modos.

También puede usar un punto de conexión de una nube privada virtual (VPC) de Amazon EC2 Auto Scaling. Los puntos de conexión de VPC permiten que los recursos de Amazon VPC utilicen sus direcciones IP privadas para acceder a Amazon EC2 Auto Scaling sin exponerse en la Internet pública. Para obtener más información, consulte Amazon EC2 Auto Scaling y puntos de conexión de VPC de tipo interfaz

Recursos relacionados

Para obtener información sobre las características para aislar el tráfico de servicios que proporciona Amazon EC2, consulte Seguridad de la infraestructura de Amazon EC2 en la Guía del usuario de Amazon EC2.

Resiliencia en Amazon EC2 Auto Scaling

La infraestructura AWS global se basa Regiones de AWS en distintas zonas de disponibilidad. Regiones de AWS proporcionan varias zonas de disponibilidad aisladas y separadas físicamente, que están conectadas mediante redes de baja latencia, alto rendimiento y alta redundancia. Con las zonas de disponibilidad, puede diseñar y utilizar aplicaciones y bases de datos que realizan una conmutación por error automática entre las zonas sin interrupciones. Las zonas de disponibilidad tienen una mayor disponibilidad, tolerancia a errores y escalabilidad que las infraestructuras tradicionales de uno o varios centros de datos.

Para obtener más información sobre las zonas de disponibilidad Regiones de AWS y las zonas de disponibilidad, consulte Infraestructura global.AWS

Para aprovechar la redundancia geográfica del diseño de la zona de disponibilidad, haga lo siguiente:

- Extienda el grupo de escalado automático en varias zonas de disponibilidad.
- Mantenga al menos una instancia en cada zona de disponibilidad.
- Conecte un equilibrador de carga para distribuir el tráfico entrante en las mismas zonas de disponibilidad. Si usa un equilibrador de carga de aplicación, asegúrese de que cada instancia de EC2 reciba una cantidad similar de tráfico; para ello, mantenga activado el equilibrador de carga entre zonas. Esto ayuda a limitar el impacto del aumento de la carga en las instancias existentes durante un evento de conmutación por error y da como resultado una mayor resiliencia que la que habría sin un equilibrio de carga entre zonas.
- Asegúrese de que las comprobaciones de estado de Elastic Load Balancing estén configuradas correctamente y también de que estén habilitadas en el grupo de escalado automático. A continuación, si una instancia no pasa la comprobación de estado, Elastic Load Balancing deja de enviarle tráfico y lo redirige a las instancias en buen estado, mientras que Amazon EC2 Auto Scaling sustituye a la instancia en mal estado.

Amazon EC2 Auto Scaling ayuda a dar respuesta a las necesidades de resiliencia de su aplicación de las siguientes maneras:

- Comprueba si hay problemas de estado y accesibilidad en las instancias. Cuando una instancia deja de estar en buen estado, finaliza automáticamente la instancia y lanza una nueva.
- Si hay políticas de escalado dinámico en vigor, escala automáticamente la capacidad en función del tráfico entrante.
- Detecta problemas en la confiabilidad de las CloudWatch métricas de Amazon que respaldan las políticas de escalado y detiene las actividades de escalado cuando no hay métricas confiables disponibles, por ejemplo, cuando faltan puntos de datos.
- Intenta mantener automáticamente cantidades equivalentes de instancias en cada zona de disponibilidad habilitada a medida que su grupo crece.
- Usa las zonas de disponibilidad para mantener una alta disponibilidad. Cuando una zona de disponibilidad deja de estar en buen estado, Amazon EC2 Auto Scaling hace lo siguiente:
  - Lanza nuevas instancias en distintas zonas de disponibilidad en el grupo de escalado automático.

  - Redistribuye las instancias en todas las zonas de disponibilidad habilitadas cuando la zona de disponibilidad en mal estado vuelve a un estado correcto.
- Sigue intentando lanzar instancias en otras zonas de disponibilidad habilitadas si una instancia no se lanza en una zona de disponibilidad determinada.
- Registra y anula automáticamente las instancias con los equilibradores de carga asociados al grupo de escalado automático. De esta forma, no es necesario registrar y anular por separado el registro de las instancias.
- Las interrupciones en el plano de control del APIs servicio Auto Scaling de Amazon EC2 no afectarán al escalado de los grupos de Auto Scaling existentes.

Recursos relacionados

Para obtener información sobre las características que ayudan a respaldar sus necesidades de resiliencia de datos proporcionada por Amazon EBS, consulte Resiliencia en Amazon Elastic Block Store en la Guía del usuario de Amazon EBS.

Protección de los datos en Amazon EC2 Auto Scaling

El modelo de se aplica a la protección de datos en Amazon EC2 Auto Scaling. Como se describe en este modelo, AWS es responsable de proteger la infraestructura global en la que se ejecutan todos los Nube de AWS. Eres responsable de mantener el control sobre el contenido alojado en esta infraestructura. También eres responsable de las tareas de administración y configuración de seguridad para los Servicios de AWS que utiliza. Para obtener más información sobre la privacidad de los datos, consulte las Preguntas frecuentes sobre privacidad de datos y los . Para obtener información sobre la protección de datos en Europa, consulte el Centro del Reglamento General de Protección de Datos (GDPR).

Para proteger los datos, le recomendamos que proteja Cuenta de AWS las credenciales y configure los usuarios individuales con AWS IAM Identity Center o AWS Identity and Access Management (IAM). De esta manera, solo se otorgan a cada usuario los permisos necesarios para cumplir sus obligaciones laborales. También recomendamos proteger sus datos de la siguiente manera:

- Utiliza la autenticación multifactor (MFA) en cada cuenta.
- Se utiliza SSL/TLS para comunicarse con AWS los recursos. Exigimos TLS 1.2 y recomendamos TLS 1.3.

- Configure la API y el registro de actividad de los usuarios con AWS CloudTrail. Para obtener información sobre el uso de CloudTrail senderos para capturar AWS actividades, consulte Cómo trabajar con CloudTrail senderos en la Guía del AWS CloudTrail usuario.
- Utilice soluciones de AWS cifrado, junto con todos los controles de seguridad predeterminados Servicios de AWS.
- Utiliza servicios de seguridad administrados avanzados, como Amazon Macie, que lo ayuden a detectar y proteger la información confidencial almacenada en Amazon S3.
- Si necesita módulos criptográficos validados por FIPS 140-3 para acceder a AWS través de una interfaz de línea de comandos o una API, utilice un punto final FIPS. Para obtener más información sobre los puntos de conexión de FIPS disponibles, consulte Estándar de procesamiento de la información federal (FIPS) 140-3.

Se recomienda encarecidamente no introducir nunca información confidencial o sensible, como por ejemplo, direcciones de correo electrónico de clientes, en etiquetas o campos de formato libre, tales como el campo Nombre. Esto incluye cuando trabaja con Amazon EC2 Auto Scaling u Servicios de AWS otro dispositivo mediante la consola, la API AWS CLI AWS o los SDK. Cualquier dato que introduzca en etiquetas o campos de formato libre utilizados para los nombres se pueden emplear para los registros de facturación o diagnóstico. Si proporciona una URL a un servidor externo, recomendamos encarecidamente que no incluya información de credenciales en la URL a fin de validar la solicitud para ese servidor.

Cuando se inicializa una instancia de Amazon EC2, tiene la opción de transferir los datos de usuario a la instancia para hacer configuraciones adicionales cuando la instancia arranca. También le recomendamos que nunca incluya información confidencial o sensible en los datos del usuario que se transferirán a una instancia.

Úselo AWS KMS keys para cifrar volúmenes de Amazon EBS

Puede configurar su grupo de Auto Scaling para cifrar los datos de volúmenes de Amazon EBS almacenados en la nube con AWS KMS keys. Amazon EC2 Auto Scaling AWS admite claves administradas y administradas por el cliente para cifrar los datos. Tenga en cuenta que la opción `KmsKeyId` de especificar una clave administrada por el cliente no está disponible cuando se utiliza una configuración de lanzamiento. Para especificar la clave administrada por el cliente, utilice una plantilla de lanzamiento en su lugar. Para obtener más información, consulte Creación de una plantilla de lanzamiento para un grupo de Auto Scaling. Para obtener información sobre cómo crear, almacenar y administrar sus claves de AWS KMS cifrado, consulte la Guía para AWS Key Management Service desarrolladores.

También puede configurar una clave gestionada por el cliente en su EBS-backed AMI antes de configurar la plantilla de lanzamiento o la configuración de lanzamiento, o utilizar el cifrado de forma predeterminada para aplicar el cifrado de los nuevos volúmenes de EBS y copias instantáneas que cree. Para obtener más información, consulte Uso del cifrado con las EBS-backed AMI en la Guía del usuario de Amazon EC2 y Cifrado predeterminado en la Guía del usuario de Amazon EBS.

> **Note**
>
> Para obtener información acerca de cómo configurar la política de claves que necesita para iniciar instancias de Auto Scaling cuando utiliza una clave administrada por el cliente para el cifrado, consulte Política de AWS KMS claves obligatoria para su uso con volúmenes cifrados.

Recursos relacionados

Para ver las directrices de protección de datos proporcionadas por Amazon EBS, consulte Protección de datos en Amazon Elastic Block Store en la Guía del usuario de Amazon EBS.

Política de AWS KMS claves obligatoria para su uso con volúmenes cifrados

Amazon EC2 Auto Scaling utiliza funciones vinculadas a servicios para delegar permisos a otras personas. Servicios de AWS Las funciones vinculadas al servicio Amazon EC2 Auto Scaling están predefinidas e incluyen los permisos que Amazon EC2 Auto Scaling necesita para llamar a Servicios de AWS otras personas en su nombre. Los permisos predefinidos también incluyen el acceso a sus. Claves administradas por AWS Sin embargo, no incluyen el acceso a las claves administradas por el cliente, lo que le permite mantener un control total de estas claves.

En este tema se describe cómo configurar la política de claves que necesita para iniciar instancias de Auto Scaling cuando especifica una clave administrada por el cliente para el cifrado de Amazon EBS.

> **Note**
>
> Amazon EC2 Auto Scaling no necesita autorización adicional para poder utilizar la Clave administrada de AWS predeterminada para proteger los volúmenes cifrados en su cuenta.

Contenido

- Descripción general de
- Configuración de las políticas de claves
- Ejemplo 1: secciones de la política de claves que permiten el acceso a la clave administrada por el cliente
- Ejemplo 2: secciones de la política de claves que permiten el acceso entre cuentas a la clave administrada por el cliente
- Edita las políticas clave en la consola AWS KMS

Descripción general de

AWS KMS keys Se puede usar lo siguiente para el cifrado de Amazon EBS cuando Amazon EC2 Auto Scaling lanza instancias:

- Clave administrada de AWS: una clave de cifrado de su cuenta que Amazon EBS crea, posee y administra. Esta es la clave de cifrado predeterminada en las cuentas nuevas. Clave administrada de AWS Se usa para el cifrado, a menos que especifique una clave administrada por el cliente.
- Clave administrada por el cliente: una clave de cifrado personalizada que usted crea, posee y administra. Para obtener más información, consulte Creación de claves en la Guía para desarrolladores de AWS Key Management Service .

  Nota: La clave debe ser simétrica. Amazon EBS no es compatible con claves asimétricas administradas por el cliente.

Puede configurar las claves administradas por el cliente al crear instantáneas cifradas o una plantilla de lanzamiento que especifique volúmenes cifrados, o habilitar el cifrado de forma predeterminada.

Configuración de las políticas de claves

Las claves de KMS deben tener una política de claves que permita que Amazon EC2 Auto Scaling inicie instancias con volúmenes de Amazon EBS cifrados con una clave administrada por el cliente.

Utilice los ejemplos de esta página para configurar una política de claves que proporcione a Amazon EC2 Auto Scaling acceso a la clave administrada por el cliente. Puede modificar la política de claves de la clave administrada por el cliente, o bien cuando se cree la clave, o bien en un momento posterior.

Como mínimo, debe agregar dos instrucciones a la política de claves para que funcione con Amazon EC2 Auto Scaling.

- La primera instrucción permite a la identidad de IAM especificada en el elemento `Principal` utilizar directamente la clave administrada por el cliente. Incluye permisos para realizar las `DescribeKey` operaciones AWS KMS `Encrypt` `DecryptReEncrypt*`,`GenerateDataKey*`, y en la clave.
- La segunda sentencia permite que la identidad de IAM especificada en el `Principal` elemento utilice la `CreateGrant` operación para generar concesiones que deleguen un subconjunto de sus propios permisos en uno de los Servicios de AWS que estén integrados con AWS KMS o con otro principal. Esto les permite utilizar la clave para crear recursos cifrados en su nombre.

Cuando agregue las nuevas instrucciones a la política de claves, no cambie las instrucciones existentes en la política.

En cada uno de los siguientes ejemplos, los argumentos que se deben reemplazar, como un identificador clave o el nombre de un rol vinculado a un servicio, se muestran como. *`user placeholder text`* En la mayoría de los casos, puede sustituir el nombre del rol vinculado a servicios por el de otro rol vinculado a servicios de Amazon EC2 Auto Scaling.

Para obtener más información, consulte los siguientes recursos:

- Para crear una clave con AWS CLI, consulte create-key.
- Para actualizar una política clave con la AWS CLI, consulta put-key-policy.
- Para encontrar el ID y el nombre de recurso de Amazon (ARN) de una clave, consulte Encontrar el ID y el ARN de la clave en la Guía para desarrolladores de AWS Key Management Service .
- Para obtener más información acerca de los roles vinculados a servicios de Amazon EC2 Auto Scaling, consulte Roles vinculados a servicios de Amazon EC2 Auto Scaling.
- Para obtener más información acerca del cifrado de Amazon EBS y KMS en general, consulte Cifrado de Amazon EBS en la Guía del usuario de Amazon EBS y la Guía para desarrolladores de AWS Key Management Service.

Ejemplo 1: secciones de la política de claves que permiten el acceso a la clave administrada por el cliente

Agregue las dos instrucciones siguientes a la política de claves de la clave administrada por el cliente, sustituyendo el ARN del ejemplo por el ARN del rol vinculado a servicios correspondiente que tiene permitido el acceso a la clave. En este ejemplo, las secciones de la política conceden al rol vinculado a servicios llamado AWSServiceRoleForAutoScaling permisos para utilizar la clave administrada por el cliente.

```
{
   "Sid": "Allow service-linked role use of the customer managed key",
   "Effect": "Allow",
   "Principal": {
       "AWS": [
           "arn:aws:iam::account-id:role/aws-service-role/
autoscaling.amazonaws.com/AWSServiceRoleForAutoScaling"
       ]
   },
   "Action": [
       "kms:Encrypt",
       "kms:Decrypt",
       "kms:ReEncrypt*",
       "kms:GenerateDataKey*",
       "kms:DescribeKey"
   ],
   "Resource": "*"
}
```

```
{
   "Sid": "Allow attachment of persistent resources",
   "Effect": "Allow",
   "Principal": {
       "AWS": [
           "arn:aws:iam::account-id:role/aws-service-role/
autoscaling.amazonaws.com/AWSServiceRoleForAutoScaling"
       ]
   },
   "Action": [
       "kms:CreateGrant"
   ],
   "Resource": "*",
   "Condition": {
```

```
        "Bool": {
            "kms:GrantIsForAWSResource": true
        }
    }
}
```

Ejemplo 2: secciones de la política de claves que permiten el acceso entre cuentas a la clave administrada por el cliente

Si crea una clave administrada por el cliente en una cuenta diferente a la del grupo de escalado automático, debe utilizar una concesión en combinación con la política de claves para permitir el acceso entre cuentas a la clave.

Hay dos pasos que deben completarse en el siguiente orden:

1. En primer lugar, agregue las dos instrucciones de política siguientes a la política clave de la clave administrada por el cliente. Sustituya el ARN de ejemplo por el ARN de la otra cuenta y asegúrese de *111122223333* reemplazarlo por el ID de cuenta real en el Cuenta de AWS que desea crear el grupo de Auto Scaling. Esto le permite otorgar a un usuario o rol de IAM en la cuenta especificada permiso para crear una concesión para la clave mediante el siguiente comando CLI. Sin embargo, esto por sí solo no otorga acceso a la clave a ningún usuario.

```
{
   "Sid": "Allow external account 111122223333 use of the customer managed key",
   "Effect": "Allow",
   "Principal": {
       "AWS": [
           "arn:aws:iam::111122223333:root"
       ]
   },
   "Action": [
       "kms:Encrypt",
       "kms:Decrypt",
       "kms:ReEncrypt*",
       "kms:GenerateDataKey*",
       "kms:DescribeKey"
   ],
   "Resource": "*"
}
```

```
{
```

```
    "Sid": "Allow attachment of persistent resources in external
 account 111122223333",
    "Effect": "Allow",
    "Principal": {
        "AWS": [
            "arn:aws:iam::111122223333:root"
        ]
    },
    "Action": [
        "kms:CreateGrant"
    ],
    "Resource": "*"
}
```

2. Luego, desde la cuenta en la que desea crear el grupo de escalado automático, cree una concesión que delegue los permisos relevantes al rol vinculado al servicio adecuado. El elemento `Grantee Principal` de la concesión es el ARN del rol vinculado al servicio pertinente. El `key-id` es el ARN de la clave.

   A continuación, se muestra un ejemplo de comando CLI create-grant que otorga *111122223333* permisos al rol vinculado al servicio mencionado AWSServiceRoleForAutoScalingen la cuenta para usar la clave administrada por el cliente en la cuenta. *444455556666*

```
aws kms create-grant \
  --region us-west-2 \
  --key-id arn:aws:kms:us-
west-2:444455556666:key/1a2b3c4d-5e6f-1a2b-3c4d-5e6f1a2b3c4d \
  --grantee-principal arn:aws:iam::111122223333:role/aws-service-role/
autoscaling.amazonaws.com/AWSServiceRoleForAutoScaling \
  --operations "Encrypt" "Decrypt" "ReEncryptFrom" "ReEncryptTo" "GenerateDataKey"
 "GenerateDataKeyWithoutPlaintext" "DescribeKey" "CreateGrant"
```

   Para que este comando se ejecute correctamente, el usuario que realiza la solicitud debe tener permisos para la acción `CreateGrant`.

   El siguiente ejemplo de política de IAM permite que una identidad de IAM (usuario o rol) en la cuenta cree una concesión *111122223333* para la cuenta clave gestionada por el cliente. *444455556666*

   JSON

```
{
```

```
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "AllowCreationOfGrantForTheKMSKeyinExternalAccount444455556666",
      "Effect": "Allow",
      "Action": "kms:CreateGrant",
      "Resource": "arn:aws:kms:us-
west-2:444455556666:key/1a2b3c4d-5e6f-1a2b-3c4d-5e6f1a2b3c4d"
    }
  ]
}
```

Para obtener más información acerca de cómo crear una concesión para una clave KMS en una Cuenta de AWS diferente, consulte Concesiones en AWS KMS en laAWS Key Management Service Guía para desarrolladores.

> **Important**
>
> El nombre del rol vinculado al servicio especificado como principal del beneficiario debe ser el nombre de un rol existente. Tras crear la concesión, para garantizar que la concesión permita que Amazon EC2 Auto Scaling utilice la clave de KMS especificada, no elimine ni vuelva a crear el rol vinculado al servicio.

Edita las políticas clave en la consola AWS KMS

En los ejemplos que aparecen en las secciones anteriores, solo se explica cómo agregar instrucciones a una política de claves, que es una de las múltiples formas de cambiar una de dichas políticas. La forma más sencilla de cambiar una política clave consiste en utilizar la vista predeterminada de la AWS KMS consola para las políticas clave y convertir una identidad de IAM (usuario o rol) en uno de los usuarios clave de la política clave correspondiente. Para obtener más información, consulte Uso de la vista Consola de administración de AWS predeterminada en la Guía para AWS Key Management Service desarrolladores.

> **Important**
>
> Tenga cuidado. Las declaraciones de política de visualización predeterminadas de la consola incluyen permisos para realizar AWS KMS `Revoke` operaciones en la clave gestionada por el cliente. Si concedes Cuenta de AWS acceso a una clave gestionada por el cliente de tu

cuenta y revocas accidentalmente la concesión que les concedió este permiso, los usuarios externos ya no podrán acceder a sus datos cifrados ni a la clave que se utilizó para cifrarlos.

Identity and Access Management para Amazon EC2 Auto Scaling

AWS Identity and Access Management (IAM) es una Servicio de AWS que ayuda al administrador a controlar de forma segura el acceso a AWS los recursos. Los administradores de IAM controlan quién está autenticado (ha iniciado sesión) y autorizado (tiene permisos) para utilizar recursos de Amazon EC2 Auto Scaling. La IAM es una Servicio de AWS herramienta que puede utilizar sin coste adicional.

Para utilizar Amazon EC2 Auto Scaling, necesita Cuenta de AWS una y sus credenciales de seguridad para iniciar sesión en su cuenta. Para obtener más información, consulte Credenciales de seguridad de AWS en la Guía del usuario de IAM.

Para ver la documentación completa de IAM, consulte la Guía del usuario de IAM.

Control de acceso

Aunque disponga de credenciales válidas para autenticar las solicitudes, si no tiene permisos, no podrá crear recursos de Amazon EC2 Auto Scaling ni acceder a ellos. Por ejemplo, debe tener permisos para crear grupos de escalado automático, lanzar instancias con plantillas de lanzamiento, etc.

En las secciones siguientes se incluyen detalles sobre cómo un administrador de IAM puede utilizar IAM para proteger sus recursos de Amazon EC2 Auto Scaling controlando quién puede realizar acciones de Amazon EC2 Auto Scaling.

Le recomendamos que lea primero los temas de Amazon EC2. Consulte Identity and Access Management para Amazon EC2 en la Guía del usuario de Amazon EC2. Después de leer los temas de esta sección, debería tener una buena idea de qué permisos de control de acceso ofrece Amazon EC2 y cómo pueden encajar con los permisos de recursos de Amazon EC2 Auto Scaling.

Temas

- Cómo funciona Amazon EC2 Auto Scaling con IAM

- Permisos de API para Amazon EC2 Auto Scaling
- AWS políticas gestionadas para Amazon EC2 Auto Scaling
- Roles vinculados a servicios de Amazon EC2 Auto Scaling
- Ejemplos de políticas basadas en identidades de Amazon EC2 Auto Scaling
- Prevención de la sustitución confusa entre servicios
- Control del uso de la plantilla de inicialización de Amazon EC2 en los grupos de escalado automático
- Rol de IAM para aplicaciones que se ejecuten en instancias de Amazon EC2

Cómo funciona Amazon EC2 Auto Scaling con IAM

Antes de utilizar IAM para administrar el acceso a Amazon EC2 Auto Scaling, debe conocer qué características de IAM se encuentran disponibles con Amazon EC2 Auto Scaling.

Funciones de IAM que puede utilizar con Amazon EC2 Auto Scaling

Característica de IAM	Compatibilidad con Amazon EC2 Auto Scaling
Políticas basadas en identidades	Sí
Políticas basadas en recursos	No
Acciones de políticas	Sí
Recursos de políticas	Sí
Claves de condición de política (específicas del servicio)	Sí
ACLs	No
ABAC (etiquetas en políticas)	Parcial
Credenciales temporales	Sí
Roles de servicio	Sí
Roles vinculados al servicio	Sí

Para obtener una visión general de cómo Amazon EC2 Auto Scaling y Servicios de AWS otros funcionan con la mayoría de las funciones de IAM, Servicios de AWS consulte Cómo funcionan con IAM en la Guía del usuario de IAM.

Políticas basadas en identidades de Amazon EC2 Auto Scaling

Compatibilidad con las políticas basadas en identidad: sí

Las políticas basadas en identidad son documentos de políticas de permisos JSON que puede asociar a una identidad, como un usuario de IAM, un grupo de usuarios o un rol. Estas políticas controlan qué acciones pueden realizar los usuarios y los roles, en qué recursos y en qué condiciones. Para obtener más información sobre cómo crear una política basada en la identidad, consulte Definición de permisos de IAM personalizados con políticas administradas por el cliente en la Guía del usuario de IAM.

Con las políticas basadas en identidades de IAM, puede especificar las acciones y los recursos permitidos o denegados, así como las condiciones en las que se permiten o deniegan las acciones. Para obtener más información sobre los elementos que puede utilizar en una política de JSON, consulte Referencia de los elementos de la política de JSON de IAM en la Guía del usuario de IAM.

Políticas basadas en recursos de Amazon EC2 Auto Scaling

Admite políticas basadas en recursos: no

Las políticas basadas en recursos son documentos de política JSON que se asocian a un recurso. Los ejemplos de políticas basadas en recursos son las políticas de confianza de roles de IAM y las políticas de bucket de Amazon S3. En los servicios que admiten políticas basadas en recursos, los administradores de servicios pueden utilizarlos para controlar el acceso a un recurso específico. Para el recurso al que se asocia la política, la política define qué acciones puede realizar una entidad principal especificada en ese recurso y en qué condiciones. Debe especificar una entidad principal en una política basada en recursos. Los principales pueden incluir cuentas, usuarios, roles, usuarios federados o. Servicios de AWS

Para habilitar el acceso entre cuentas, puede especificar toda una cuenta o entidades de IAM de otra cuenta como la entidad principal de una política en función de recursos. Para obtener más información, consulte Acceso a recursos entre cuentas en IAM en la Guía del usuario de IAM.

Acciones de política para Amazon EC2 Auto Scaling

Compatibilidad con las acciones de políticas: sí

Los administradores pueden usar las políticas de AWS JSON para especificar quién tiene acceso a qué. Es decir, qué entidad principal puede realizar acciones en qué recursos y en qué condiciones.

El elemento `Action` de una política JSON describe las acciones que puede utilizar para conceder o denegar el acceso en una política. Incluya acciones en una política para conceder permisos y así llevar a cabo la operación asociada.

Para ver una lista de las acciones de Amazon EC2 Auto Scaling, consulte Acciones definidas por Amazon EC2 Auto Scaling en Referencia de autorizaciones de servicio).

Las acciones de políticas de Amazon EC2 Auto Scaling utilizan el siguiente prefijo antes de la acción:

```
autoscaling
```

Para especificar varias acciones en una única instrucción, sepárelas con comas.

```
"Action": [
  "autoscaling:action1",
  "autoscaling:action2"
]
```

Puede utilizar caracteres comodín (*) para especificar varias acciones. Por ejemplo, para especificar todas las acciones que comiencen con la palabra `Describe`, incluya la siguiente acción:

```
"Action": "autoscaling:Describe*"
```

Recursos de políticas de Amazon EC2 Auto Scaling

Compatibilidad con los recursos de políticas: sí

Los administradores pueden usar las políticas de AWS JSON para especificar quién tiene acceso a qué. Es decir, qué entidad principal puede realizar acciones en qué recursos y en qué condiciones.

El elemento `Resource` de la política JSON especifica el objeto u objetos a los que se aplica la acción. Como práctica recomendada, especifique un recurso utilizando el Nombre de recurso de Amazon (ARN). En el caso de las acciones que no admiten permisos por recurso, utilice un carácter comodín (*) para indicar que la instrucción se aplica a todos los recursos.

```
"Resource": "*"
```

Se puede utilizar ARNs para identificar los grupos de Auto Scaling y lanzar las configuraciones a las que se aplica la política de IAM.

Un grupo de Auto Scaling tiene el siguiente ARN.

```
"Resource": "arn:aws:autoscaling:region:account-
id:autoScalingGroup:uuid:autoScalingGroupName/asg-name"
```

Una configuración de lanzamiento tiene el siguiente ARN.

```
"Resource": "arn:aws:autoscaling:region:account-
id:launchConfiguration:uuid:launchConfigurationName/lc-name"
```

Para especificar un grupo de escalado automático con la acción `CreateAutoScalingGroup`, debe sustituir el UUID por un comodín (*) como se indica a continuación.

```
"Resource": "arn:aws:autoscaling:region:account-
id:autoScalingGroup:*:autoScalingGroupName/asg-name"
```

Para especificar una configuración de lanzamiento con la acción `CreateLaunchConfiguration`, debe sustituir el UUID por un comodín (*) como se indica a continuación.

```
"Resource": "arn:aws:autoscaling:region:account-
id:launchConfiguration:*:launchConfigurationName/lc-name"
```

Para obtener más información sobre los tipos de recursos de Amazon EC2 Auto Scaling y ARNs sus tipos de recursos, consulte Recursos definidos por Amazon EC2 Auto Scaling en la Referencia de autorización de servicios. Para obtener información sobre las acciones con las que puede especificar el ARN de cada recurso, consulte Actions defined by Amazon EC2 Auto Scaling (Acciones definidas por Amazon EC2 Auto Scaling).

> **Note**
>
> Para ver un ejemplo de una política de IAM que se utiliza ARNs para controlar el acceso a los grupos de Auto Scaling, consulteControl de los grupos de escalado automático que se pueden eliminar.

No todas las acciones de Amazon EC2 Auto Scaling admiten permisos de recursos. Para las acciones que no admiten permisos de recursos, debe utilizar un comodín (*) como recurso.

Las siguientes acciones de Amazon EC2 Auto Scaling no admiten permisos de recursos.

- `DescribeAccountLimits`
- `DescribeAdjustmentTypes`
- `DescribeAutoScalingGroups`
- `DescribeAutoScalingInstances`
- `DescribeAutoScalingNotificationTypes`
- `DescribeInstanceRefreshes`
- `DescribeLaunchConfigurations`
- `DescribeLifecycleHooks`
- `DescribeLifecycleHookTypes`
- `DescribeLoadBalancers`
- `DescribeLoadBalancerTargetGroups`
- `DescribeMetricCollectionTypes`
- `DescribeNotificationConfigurations`
- `DescribePolicies`
- `DescribeScalingActivities`
- `DescribeScalingProcessTypes`
- `DescribeScheduledActions`
- `DescribeTags`
- `DescribeTerminationPolicyTypes`
- `DescribeWarmPool`

Claves de condición de políticas Amazon EC2 Auto Scaling

Compatibilidad con claves de condición de políticas específicas del servicio: sí

Los administradores pueden usar las políticas de AWS JSON para especificar quién tiene acceso a qué. Es decir, qué entidad principal puede realizar acciones en qué recursos y en qué condiciones.

El elemento `Condition` especifica cuándo se ejecutan las instrucciones en función de criterios definidos. Puede crear expresiones condicionales que utilizan operadores de condición, tales como igual o menor que, para que la condición de la política coincida con los valores de la solicitud. Para ver todas las claves de condición AWS globales, consulte las claves de contexto de condición AWS globales en la Guía del usuario de IAM.

Amazon EC2 Auto Scaling admite las siguientes claves de condición que se pueden utilizar para controlar el acceso a las acciones compatibles e imponer la configuración de los grupos de escalado automático:

- `autoscaling:InstanceTypes`
- `autoscaling:LaunchConfigurationName`
- `autoscaling:LaunchTemplateVersionSpecified`
- `autoscaling:LoadBalancerNames`
- `autoscaling:MaxSize`
- `autoscaling:MinSize`
- `autoscaling:ResourceTag/`*`key-name`*`: `*`tag-value`*
- `autoscaling:TargetGroupARNs`
- `autoscaling:VPCZoneIdentifiers`
- `autoscaling:ForceDelete`

Las claves de condición siguientes son específicas para crear solicitudes de configuración de lanzamiento:

- `autoscaling:ImageId`
- `autoscaling:InstanceType`
- `autoscaling:MetadataHttpEndpoint`
- `autoscaling:MetadataHttpPutResponseHopLimit`
- `autoscaling:MetadataHttpTokens`
- `autoscaling:SpotPrice`

Además, Amazon EC2 Auto Scaling admite las siguientes claves de condición globales que puede utilizar para definir los permisos en función de las etiquetas de la solicitud o presentar en el grupo de

escalado automático. Para obtener más información, consulte Etiquetado de grupos e instancias de Auto Scaling.

- `aws:RequestTag/key-name: tag-value`
- `aws:ResourceTag/key-name: tag-value`
- `aws:TagKeys: [tag-key, ...]`

Para conocer las acciones de API de Amazon EC2 Auto Scaling que pueda usar con una clave de condición, consulte Acciones definidas por Amazon EC2 Auto Scaling en la Referencia de autorizaciones de servicio. Para obtener más información sobre las claves de condición de Amazon EC2 Auto Scaling, consulte Claves de condición de Amazon EC2 Auto Scaling.

> **Note**
>
> Para ver ejemplos de políticas de IAM que utilizan claves de condición para controlar el acceso a las acciones compatibles e imponer la configuración de los grupos de escalado automático, consulte los siguientes recursos:
>
> - Requerimiento de una plantilla de lanzamiento y un número de versión: en este ejemplo se exige que se especifique una plantilla de inicialización y el número de versión de la misma al crear o actualizar los grupos de escalado automático.
> - Control del tamaño de los grupos de escalado automático que se pueden crear: en este ejemplo se imponen restricciones a los valores posibles para las propiedades `MinSize` y `MaxSize` al crear o actualizar grupos de escalado automático con una etiqueta específica.
> - Controlar qué políticas de escalado se pueden eliminar: en este ejemplo se exige que la eliminación de políticas de escalado solo esté permitida para los grupos de escalado automático sin una etiqueta específica.

ACL en Amazon EC2 Auto Scaling

Soportes ACLs: No

Las listas de control de acceso (ACLs) controlan qué directores (miembros de la cuenta, usuarios o roles) tienen permisos para acceder a un recurso. ACLs son similares a las políticas basadas en recursos, aunque no utilizan el formato de documento de políticas JSON.

ABAC con Amazon EC2 Auto Scaling

Compatibilidad con ABAC (etiquetas en las políticas): parcial

El control de acceso basado en atributos (ABAC) es una estrategia de autorización que define permisos en función de atributos denominados etiquetas. Puede adjuntar etiquetas a las entidades y AWS los recursos de IAM y, a continuación, diseñar políticas de ABAC para permitir las operaciones cuando la etiqueta del director coincida con la etiqueta del recurso.

Para controlar el acceso en función de etiquetas, debe proporcionar información de las etiquetas en el elemento de condición de una política utilizando las claves de condición `aws:ResourceTag/key-name`, `aws:RequestTag/key-name` o `aws:TagKeys`.

Si un servicio admite las tres claves de condición para cada tipo de recurso, el valor es Sí para el servicio. Si un servicio admite las tres claves de condición solo para algunos tipos de recursos, el valor es Parcial.

Para obtener más información sobre ABAC, consulte Definición de permisos con la autorización de ABAC en la Guía del usuario de IAM. Para ver un tutorial con los pasos para configurar ABAC, consulte Uso del control de acceso basado en atributos (ABAC) en la Guía del usuario de IAM.

ABAC es posible para los recursos que admiten etiquetas, pero no todos las admiten. Las configuraciones de lanzamiento y las políticas de escalado no admiten etiquetas, pero los grupos de escalado automático sí.

Para obtener más información, consulte Etiquetado de grupos e instancias de Auto Scaling.

Uso de credenciales temporales con Amazon EC2 Auto Scaling

Compatibilidad con credenciales temporales: sí

Las credenciales temporales proporcionan acceso a AWS los recursos a corto plazo y se crean automáticamente cuando se utiliza la federación o se cambia de rol. AWS recomienda generar credenciales temporales de forma dinámica en lugar de utilizar claves de acceso a largo plazo. Para obtener más información, consulte Credenciales de seguridad temporales en IAM y Servicios de AWS que funcionan con IAM en la Guía del usuario de IAM.

Roles de servicio para Amazon EC2 Auto Scaling

Compatible con roles de servicio: sí

Un rol de servicio es un rol de IAM que asume un servicio para realizar acciones en su nombre. Un administrador de IAM puede crear, modificar y eliminar un rol de servicio desde IAM. Para obtener más información, consulte Crear un rol para delegar permisos a un Servicio de AWS en la Guía del usuario de IAM.

Cuando crea un enlace de ciclo de vida que notifica un tema de Amazon SNS o una cola de Amazon SQS, debe especificar un rol para permitir que Amazon EC2 Auto Scaling acceda a Amazon SNS o Amazon SQS en su nombre. Usar la consola de IAM para configurar el rol de servicio para su enlace de ciclo de vida. La consola sirve para crear un rol con un conjunto suficiente de permisos mediante una política administrada. Para obtener más información, consulte Recepción de notificaciones mediante Amazon SNS y Recepción de notificaciones mediante Amazon SQS.

Al crear un grupo de Auto Scaling, si lo desea, puede transferir un rol de servicio para permitir que las instancias de Amazon EC2 accedan a otras Servicios de AWS en su nombre. El rol de servicio para instancias Amazon EC2 (también denominada perfil de instancia Amazon EC2 para una plantilla de lanzamiento o configuración de lanzamiento) es un tipo especial de función de servicio que se asigna a cada instancia de EC2 en un grupo de escalado automático cuando se lanza la instancia. Puede utilizar la consola de IAM AWS CLI para crear o editar este rol de servicio. Para obtener más información, consulte Rol de IAM para aplicaciones que se ejecuten en instancias de Amazon EC2.

> **Warning**
>
> Cambiar los permisos de un rol de servicio podría interrumpir la funcionalidad de Amazon EC2 Auto Scaling. Edite los roles de servicio solo cuando Amazon EC2 Auto Scaling proporcione orientación para hacerlo.

Roles vinculados a servicios de Amazon EC2 Auto Scaling

Compatible con roles vinculados al servicio: sí

Un rol vinculado a un servicio es un tipo de rol de servicio que está vinculado a un. Servicio de AWS El servicio puede asumir el rol para realizar una acción en su nombre. Los roles vinculados al servicio aparecen en usted Cuenta de AWS y son propiedad del servicio. Un administrador de IAM puede ver, pero no editar, los permisos de los roles vinculados a servicios.

Para obtener más información sobre cómo crear o administrar roles vinculados a servicios de Amazon EC2 Auto Scaling, consulte Roles vinculados a servicios de Amazon EC2 Auto Scaling.

Permisos de API para Amazon EC2 Auto Scaling

Debe conceder a los usuarios permisos para llamar a las acciones de la API de Amazon EC2 Auto Scaling que necesiten, tal y como se describe en Acciones de política para Amazon EC2 Auto Scaling. Además, para algunas acciones de Amazon EC2 Auto Scaling, debe conceder a los usuarios permiso para invocar acciones específicas desde AWS APIs otras.

Permisos necesarios de otros AWS APIs

Además de los permisos de la API Amazon EC2 Auto Scaling, los usuarios deben tener los siguientes permisos de AWS APIs otros usuarios para realizar correctamente la acción asociada.

Creación de un grupo de escalado automático (`autoscaling:CreateAutoScalingGroup`)

- `iam:CreateServiceLinkedRole`: para crear el rol vinculado al servicio predeterminado si ese rol aún no existe.
- `iam:PassRole`: para transferir un rol de IAM al servicio o a las instancias de EC2 en la inicialización. Es necesario cuando se proporciona un rol vinculado a servicios no predeterminado, un rol de IAM para un enlace de ciclo de vida o una plantilla de lanzamiento que especifique un perfil de instancia (un contenedor para un rol de IAM).
- `ec2:RunInstances`: para inicializar instancias cuando se proporciona una plantilla de inicialización.
- `ec2:CreateTags`: para etiquetar instancias y volúmenes en el momento de inicialización cuando se proporciona una plantilla de inicialización con una especificación de etiqueta.

Creación de un enlace de ciclo de vida (`autoscaling:PutLifecycleHook`)

- `iam:PassRole`: para transferir un rol de IAM al servicio. Se necesita cuando se proporciona un rol de IAM.

Asociación de un grupo de destino de VPC Lattice (`autoscaling:AttachTrafficSources`)

- `vpc-lattice:RegisterTargets`: para registrar instancias de manera automática con el grupo de destino.

Separación de un grupo de destinos de VPC Lattice (`autoscaling:DetachTrafficSources`)

- `vpc-lattice:DeregisterTargets`: para anular de manera automática el registro de instancias con el grupo de destino.

Creación de una configuración de lanzamiento (`autoscaling:CreateLaunchConfiguration`)

- `ec2:DescribeImages`
- `ec2:DescribeInstances`

- `ec2:DescribeInstanceAttribute`
- `ec2:DescribeKeyPairs`
- `ec2:DescribeSecurityGroups`
- `ec2:DescribeSpotInstanceRequests`
- `ec2:DescribeVpcClassicLink`
- `iam:PassRole`: para transferir un rol de IAM a las instancias de EC2 en la inicialización. Es necesario cuando una configuración de lanzamiento especifica un perfil de instancia (un contenedor de un rol de IAM).

AWS políticas gestionadas para Amazon EC2 Auto Scaling

Una política AWS administrada es una política independiente creada y administrada por. AWS AWS Las políticas administradas están diseñadas para proporcionar permisos para muchos casos de uso comunes, de modo que pueda empezar a asignar permisos a usuarios, grupos y funciones.

Ten en cuenta que es posible que las políticas AWS administradas no otorguen permisos con privilegios mínimos para tus casos de uso específicos, ya que están disponibles para que los usen todos los AWS clientes. Se recomienda definir políticas administradas por el cliente específicas para sus casos de uso a fin de reducir aún más los permisos.

No puedes cambiar los permisos definidos en AWS las políticas administradas. Si AWS actualiza los permisos definidos en una política AWS administrada, la actualización afecta a todas las identidades principales (usuarios, grupos y roles) a las que está asociada la política. AWS es más probable que actualice una política AWS administrada cuando Servicio de AWS se lance una nueva o cuando estén disponibles nuevas operaciones de API para los servicios existentes.

Para obtener más información, consulte Políticas administradas por AWS en la Guía del usuario de IAM.

Políticas administradas por Amazon EC2 Auto Scaling

Puede adjuntar las siguientes políticas gestionadas a sus identidades AWS Identity and Access Management (usuarios o roles) (de IAM). Cada política proporciona acceso a la totalidad o parte de las acciones de la API para Amazon EC2 Auto Scaling.

- AutoScalingConsoleFullAccess— Otorga acceso completo a Amazon EC2 Auto Scaling mediante el Consola de administración de AWS. Esta política funciona cuando se utilizan configuraciones de lanzamiento, pero no cuando se utilizan plantillas de lanzamiento.

- AutoScalingConsoleReadOnlyAccess— Otorga acceso de solo lectura a Amazon EC2 Auto Scaling mediante. Consola de administración de AWS Esta política funciona cuando se utilizan configuraciones de lanzamiento, pero no cuando se utilizan plantillas de lanzamiento.
- AutoScalingFullAccess— Otorga acceso total a Amazon EC2 Auto Scaling para las identidades de IAM que necesitan acceso total a Amazon EC2 Auto Scaling AWS CLI desde SDKs el quirófano, pero no acceso. Consola de administración de AWS
- AutoScalingReadOnlyAccess— Otorga acceso de solo lectura a Amazon EC2 Auto Scaling para las identidades de IAM que realizan llamadas únicamente al quirófano. AWS CLI SDKs

Cuando se utilizan plantillas de lanzamiento desde la consola, debe conceder permisos adicionales específicos a las plantillas de lanzamiento, tal y como se explica en Control del uso de la plantilla de inicialización de Amazon EC2 en los grupos de escalado automático. La consola de Amazon EC2 Auto Scaling necesita permisos para las acciones `ec2`, de modo que pueda mostrar información sobre las plantillas e instancias de lanzamiento mediante plantillas de lanzamiento.

AutoScalingServiceRolePolicy AWS política gestionada

Esta política se asocia a un rol vinculado al servicio que permite a Amazon EC2 Auto Scaling realizar acciones en su nombre. Para obtener más información, consulte Roles vinculados a servicios de Amazon EC2 Auto Scaling.

Para ver los permisos de esta política, consulte AutoScalingServiceRolePolicy en la Referencia de la política administrada de AWS .

Amazon EC2 Auto Scaling actualiza las políticas AWS administradas

Consulte los detalles sobre las actualizaciones de las políticas AWS administradas para Amazon EC2 Auto Scaling desde que este servicio comenzó a rastrear estos cambios. Para obtener alertas automáticas sobre cambios en esta página, suscríbase a la fuente RSS en la página Historial de documentos de Amazon EC2 Auto Scaling.

Cambio	Descripción	Fecha
Amazon EC2 Auto Scaling actualiza los permisos de su función vinculada a servicios	La `AutoScalingServiceRolePolicy` política ahora incluye el `ec2:CreateReplaceRootVolume`	18 de noviembre de 2025

Cambio	Descripción	Fecha
	`Task` permiso que le permite reemplazar los volúmenes raíz en las instancias como parte de una actualización de la instancia. Además, el `events:DescribeRule` permiso se ha eliminado de la política. Para obtener más información, consulte Roles vinculados a servicios de Amazon EC2 Auto Scaling.	
Amazon EC2 Auto Scaling añade permisos a su rol vinculado a servicios	La `AutoScalingServiceRolePolicy` política ahora incluye el permiso para activar la acción de la Grupos de recursos de AWS ListGroupResourcesAPI para obtener todos los nombres de los recursos (ARNs) de los recursos que son miembros de un grupo de recursos específico. Para obtener más información, consulte Roles vinculados a servicios de Amazon EC2 Auto Scaling.	20 de noviembre de 2024

Cambio	Descripción	Fecha
Amazon EC2 Auto Scaling añade permisos a su rol vinculado a servicios	La `AutoScalingServiceRolePolicy` política ahora concede permisos para llamar a la acción de la GetSecurityGroupsForVpcAPI de Amazon EC2 a fin de obtener todos los grupos de seguridad de una VPC para mejorar la validación, y a la acción de la GetInstanceTypesFromInstanceRequirementsAPI de Amazon EC2 para obtener información sobre los tipos de instancias que cumplen un determinado conjunto de requisitos de instancia. Para obtener más información, consulte Roles vinculados a servicios de Amazon EC2 Auto Scaling.	29 de febrero de 2024

Cambio	Descripción	Fecha
Amazon EC2 Auto Scaling añade permisos a su rol vinculado a servicios	La política `AutoScalingServiceRolePolicy` ahora otorga permisos al servicio para acceder a las acciones de la API que necesita en la integración con VPC Lattice. • Acciones `GetTargetGroup` y `ListTargetGroup`. Se necesita para recuperar información sobre los grupos de destino de VPC Lattice. • Acciones `RegisterTargets` y `DeregisterTargets`. Necesario para registrar y anular el registro de instancias de los grupos de destino de VPC Lattice. • `ListTargets`: permite que Amazon EC2 Auto Scaling recupere la información de estado de las instancias registradas en los grupos de destino de VPC Lattice. Para obtener más información, consulte Roles vinculados a servicios de Amazon EC2 Auto Scaling.	6 de diciembre de 2022

Cambio	Descripción	Fecha
Amazon EC2 Auto Scaling añade permisos a su rol vinculado a servicios	Para permitir el uso de un AWS Systems Manager parámetro como alias para un ID de AMI al crear una plantilla de lanzamiento, la `AutoScalingServiceRolePolicy` política ahora otorga permiso para llamar a la acción de la AWS Systems Manager GetParametersAPI. Para obtener más informaci ón, consulte Roles vinculado s a servicios de Amazon EC2 Auto Scaling.	28 de marzo de 2022
Amazon EC2 Auto Scaling añade permisos a su rol vinculado a servicios	Para admitir el escalado predictivo, la política `AutoScalingServiceRolePolicy` ahora permite llamar a la acción de la API CloudWatch GetMetricData. Para obtener más informaci ón, consulte Roles vinculado s a servicios de Amazon EC2 Auto Scaling.	19 de mayo de 2021
Amazon EC2 Auto Scaling comenzó a hacer el seguimien to de los cambios	Amazon EC2 Auto Scaling comenzó a realizar un seguimiento de los cambios en AWS sus políticas gestionadas.	19 de mayo de 2021

Roles vinculados a servicios de Amazon EC2 Auto Scaling

Amazon EC2 Auto Scaling utiliza roles vinculados a servicios para los permisos que necesita a fin de llamar a otros Servicios de AWS en su nombre. Un rol vinculado a un servicio es un tipo único de rol de IAM que está vinculado directamente a un. Servicio de AWS

Los roles vinculados a servicios ofrecen una manera segura de delegar permisos a otros Servicios de AWS , ya que solo los servicios vinculados pueden asumir roles vinculados a servicios. Para obtener más información, consulte Creación de un rol vinculado al servicio en la Guía del usuario de IAM. Los roles vinculados al servicio también permiten ver todas las llamadas a la API. AWS CloudTrail Esta ayuda a monitorear y auditar los requisitos, ya que se puede hacer un seguimiento de todas las acciones que Amazon EC2 Auto Scaling lleva a cabo en su nombre. Para obtener más información, consulte Registre las llamadas a la API Amazon EC2 Auto Scaling con AWS CloudTrail.

En las secciones siguientes se describe cómo crear y administrar roles vinculados a servicios de Amazon EC2 Auto Scaling. Para empezar, configure permisos que permitan a una identidad de IAM (como un usuario o rol) crear, editar o eliminar un rol vinculado al servicio.

Contenido

- Descripción general de
- Permisos concedidos por el rol vinculado a servicios
- Regiones que admiten los roles vinculados a servicios de Amazon EC2 Auto Scaling
- Creación, edición y eliminación de un rol vinculado a un servicio
  - Creación de un rol vinculado a servicios (automático)
  - Creación de un rol vinculado a servicios (manual)
  - Editar el rol vinculado a servicios
  - Eliminar el rol vinculado a servicios

Descripción general de

Hay dos tipos de roles vinculados a servicios de Amazon EC2 Auto Scaling:

- El rol vinculado a servicios predeterminado de la cuenta, denominado AWSServiceRoleForAutoScaling. Este rol se asigna automáticamente a los grupos de Auto Scaling, a menos que se haya especificado otro rol vinculado a servicios.

- Un rol vinculado a servicios con un sufijo personalizado que se especifica al crear el rol, por ejemplo, AWSServiceRoleForAutoScaling_*mysuffix*.

Los permisos de un rol vinculado a servicios con un sufijo personalizado son idénticos a los del rol vinculado a servicios predeterminado. En ninguno de los dos casos podrá editar los roles. Tampoco podrá eliminarlos si hay un grupo de escalado automático que los está usando. La única diferencia es el sufijo del nombre del rol.

Puede especificar cualquiera de las funciones al editar las políticas de AWS Key Management Service claves para permitir que las instancias lanzadas por Amazon EC2 Auto Scaling se cifren con la clave gestionada por el cliente. Sin embargo, si tiene previsto proporcionar acceso pormenorizado a una determinada clave administrada por el cliente, debe utilizar un rol vinculado a servicios con un sufijo personalizado. El uso de un rol vinculado a servicios con un sufijo personalizado le ofrece:

- Mayor control sobre la clave administrada por el cliente
- La capacidad de rastrear qué grupo de Auto Scaling realizó una llamada a la API en sus CloudTrail registros

Si crea las claves administradas por el cliente de forma que no todos los usuarios tangan acceso a ellas, siga estos pasos para permitir el uso de un rol vinculado a servicios con un sufijo personalizado:

1. Cree un rol vinculado a servicios con un sufijo personalizado. Para obtener más información, consulte Creación de un rol vinculado a servicios (manual).
2. Proporcione al rol vinculado a servicios acceso a una clave administrada por el cliente. Para obtener más información acerca de la política de claves que permite que un rol vinculado a servicios utilice la clave, consulte Política de AWS KMS claves obligatoria para su uso con volúmenes cifrados.
3. Dé acceso a los usuarios al rol vinculado a servicios que creó. Para obtener más información sobre la creación de la política de IAM, consulte Controle qué función vinculada al servicio puede transferirse (mediante) PassRole. Si los usuarios intentan especificar un rol vinculado a servicios sin permiso para transferir ese rol al servicio, recibirán un error.

Permisos concedidos por el rol vinculado a servicios

Amazon EC2 Auto Scaling utiliza el rol vinculado a servicios denominado AWSServiceRoleForAutoScaling o el rol suffix service-linked personalizado.

El rol vinculado a servicio de confía en el siguiente servicio para asumir el rol:

- `autoscaling.amazonaws.com`

La política de permisos del rol, AutoScalingServiceRolePolicy, permite que Amazon EC2 Auto Scaling complete las siguientes acciones:

- `ec2`: crear, describir, modificar, iniciar o detener y terminar instancias de EC2.
- `iam`: transmitir los roles de IAM a las instancias de EC2 para que las aplicaciones que se ejecutan en las instancias puedan acceder a las credenciales temporales del rol.
- `iam`— Cree el rol vinculado al servicio AWSServiceRoleForEC2Spot para permitir que Amazon EC2 Auto Scaling lance instancias puntuales en su nombre.
- `elasticloadbalancing`: registrar y anular el registro de instancias con Elastic Load Balancing y comprobar el estado de los destinos registrados.
- `cloudwatch`— Cree, describa, modifique y elimine CloudWatch las alarmas para las políticas de escalado y recupere las métricas utilizadas para el escalado predictivo.
- `sns`: publicar notificaciones en Amazon SNS cuando se inicialicen o terminen instancias.
- `events`— Cree, describa, actualice y elimine EventBridge reglas en su nombre.
- `ssm`: lea los parámetros del Almacén de parámetros cuando utilice un parámetro de Systems Manager como alias de un ID de AMI en una plantilla de inicialización.
- `vpc-lattice`: registrar y anular el registro de instancias con VPC Lattice y comprobar el estado de los objetivos registrados.
- `resource-groups`— Obtenga todos los nombres de los recursos (ARNs) de los recursos que son miembros de un grupo de recursos específico.

Regiones que admiten los roles vinculados a servicios de Amazon EC2 Auto Scaling

Amazon EC2 Auto Scaling admite el uso de funciones vinculadas a servicios en todos los lugares en los que Regiones de AWS el servicio esté disponible.

Creación, edición y eliminación de un rol vinculado a un servicio

Creación de un rol vinculado a servicios (automático)

Amazon EC2 Auto Scaling crea automáticamente el rol vinculado a servicios AWSServiceRoleForAutoScaling la primera vez que usted crea un grupo de escalado automático, a menos que cree manualmente un rol vinculado a servicios con sufijo personalizado y lo especifique al crear el grupo.

Debe tener permisos de IAM para crear el rol vinculado a servicios. De lo contrario, la creación automática no se lleva a cabo. Para obtener más información, consulte Permisos de roles vinculados a servicios en la Guía del usuario de IAM y Creación de un rol vinculado al servicio en esta guía.

Creación de un rol vinculado a servicios (manual)

Para crear un rol vinculado a un servicio (consola)

1. Abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, seleccione Roles, Crear rol.
3. En Select trusted entity (Seleccionar entidad de confianza), elija AWS service (Servicio de).
4. En Choose the service that will use this role (Seleccione el servicio que va a usar este rol), elija EC2 Auto Scaling y el caso de uso EC2 Auto Scaling.
5. Seleccione Next: Permissions (Siguiente: permisos), Next: Tags (Siguiente: etiquetas) y Next: Review (Siguiente: revisar). Nota: no se pueden asociar etiquetas a un rol vinculado a servicios durante su creación.
6. En la página de revisión, deje el nombre del rol en blanco para crear un rol vinculado al servicio con ese nombre AWSServiceRoleForAutoScalingo introduzca un sufijo para crear un rol vinculado al servicio con el nombre _. AWSServiceRoleForAutoScaling ***suffix***
7. (Opcional) En Role description (Descripción del rol), modifique la descripción del nuevo rol vinculado a servicios.
8. Elija Crear rol.

Para crear un rol vinculado a un servicio (AWS CLI)

Utilice el siguiente comando create-service-linked-roleCLI para crear un rol vinculado a un servicio para Amazon EC2 Auto Scaling con el nombre _. AWSServiceRoleForAutoScaling ***suffix***

```
aws iam create-service-linked-role --aws-service-name autoscaling.amazonaws.com --
custom-suffix suffix
```

El resultado de este comando incluye el ARN del rol vinculado a servicios, que puede utilizar para conceder a este rol acceso a su clave administrada por el cliente.

```
{
    "Role": {
        "RoleId": "ABCDEF0123456789ABCDEF",
        "CreateDate": "2018-08-30T21:59:18Z",
        "RoleName": "AWSServiceRoleForAutoScaling_suffix",
        "Arn": "arn:aws:iam::123456789012:role/aws-service-role/
autoscaling.amazonaws.com/AWSServiceRoleForAutoScaling_suffix",
        "Path": "/aws-service-role/autoscaling.amazonaws.com/",
        "AssumeRolePolicyDocument": {
            "Version": "2012-10-17",
            "Statement": [
                {
                    "Action": [
                        "sts:AssumeRole"
                    ],
                    "Principal": {
                        "Service": [
                            "autoscaling.amazonaws.com"
                        ]
                    },
                    "Effect": "Allow"
                }
            ]
        }
    }
}
```

Para obtener más información, consulte Creación de un rol vinculado al servicio en la Guía del usuario de IAM.

Editar el rol vinculado a servicios

Los roles vinculados a servicios que se crean para Amazon EC2 Auto Scaling no se pueden editar. Después de crear un rol vinculado a servicios, no se puede modificar el nombre ni los permisos. Sin embargo, sí se puede modificar la descripción del rol. Para obtener más información, consulte la Descripción sobre cómo editar un rol vinculado al servicio en la Guía del usuario de IAM.

Eliminar el rol vinculado a servicios

Si no está utilizando un grupo de escalado automático, le recomendamos que elimine el rol vinculado a servicios. Si lo elimina, evitará tener una entidad que no se utiliza o no tendrá que monitorizarla o mantenerla de forma activa.

Solo puede eliminar un rol vinculado a servicios después de eliminar los recursos dependientes relacionados. De este modo, evitará también que pueda revocar por accidente los permisos de Amazon EC2 Auto Scaling para los recursos. Si un rol vinculado a servicios se utiliza con varios grupos de Auto Scaling, debe eliminar todos los grupos de Auto Scaling que utilicen ese rol vinculado a servicios para poder eliminarlo. Para obtener más información, consulte Eliminación de la infraestructura de Auto Scaling.

Puede utilizar IAM para eliminar un rol vinculado a servicios. Para obtener más información, consulte Eliminación de un rol vinculado a servicios en la Guía del usuario de IAM.

Si elimina el rol vinculado a servicios AWSServiceRoleForAutoScaling, Amazon EC2 Auto Scaling lo volverá a crear cuando usted cree un grupo de escalado automático, pero no especifique otro rol vinculado a servicios.

Ejemplos de políticas basadas en identidades de Amazon EC2 Auto Scaling

De forma predeterminada, un usuario nuevo no Cuenta de AWS tiene permisos para hacer nada. Un administrador de IAM debe crear y asignar políticas de IAM que concedan permiso de identidad de IAM (como a los usuarios o roles) para realizar acciones de API de Amazon EC2 Auto Scaling.

Para obtener más información acerca de cómo crear una política de IAM con estos documentos de políticas de JSON de ejemplo, consulte Creación de políticas en la pestaña JSON en la Guía del usuario de IAM.

A continuación se muestra un ejemplo de una política de permisos.

JSON

```
{
   "Version":"2012-10-17",
   "Statement": [{
      "Effect": "Allow",
      "Action": [
          "autoscaling:CreateAutoScalingGroup",
          "autoscaling:UpdateAutoScalingGroup",
```

```
            "autoscaling:DeleteAutoScalingGroup"
        ],
        "Resource": "*",
        "Condition": {
            "StringEquals": { "autoscaling:ResourceTag/purpose": "testing" }
        }
    },
    {
        "Effect": "Allow",
        "Action": "autoscaling:Describe*",
        "Resource": "*"
    }]
}
```

Esta política de ejemplo proporciona a los usuarios permisos para crear, modificar y eliminar grupos de escalado automático, pero solo si el grupo utiliza la etiqueta **`purpose=testing`**. Como las acciones `Describe` no admiten permisos de nivel de recursos, debe especificarlas en una instrucción aparte sin condiciones. Para lanzar instancias con una plantilla de lanzamiento, el usuario también debe tener el permiso `ec2:RunInstances`. Para obtener más información, consulte Control del uso de la plantilla de inicialización de Amazon EC2 en los grupos de escalado automático.

> **Note**
>
> Puede crear sus propias políticas de IAM personalizadas para permitir o denegar los permisos de las identidades de IAM (grupos o roles) para llevar a cabo acciones de Amazon EC2 Auto Scaling. Puede conectar estas políticas personalizadas a las identidades de IAM que necesiten los permisos especificados. A continuación se muestran algunos ejemplos de permisos para algunos casos de uso comunes.
>
> Algunas acciones de la API de Amazon EC2 Auto Scaling le permiten incluir en la política grupos de Auto Scaling específicos que la acción puede crear o modificar. Puede restringir los recursos de destino para estas acciones especificando un grupo de Auto Scaling individual ARNs. Sin embargo, como práctica recomendada, se sugiere utilizar políticas basadas en etiquetas que permitan o denieguen acciones en grupos de Auto Scaling con una etiqueta específica.

Ejemplos

- Control del tamaño de los grupos de escalado automático que se pueden crear
- Controlar qué claves de etiqueta y valores de etiqueta se pueden utilizar
- Control de los grupos de escalado automático que se pueden eliminar
- Controlar qué políticas de escalado se pueden eliminar
- Control del acceso a las acciones de actualización de instancias
- Creación de un rol vinculado al servicio
- Controle qué función vinculada al servicio puede transferirse (mediante) PassRole

Control del tamaño de los grupos de escalado automático que se pueden crear

La siguiente política concede a los usuarios permisos para crear y actualizar todos los grupos de escalado automático con la etiqueta **`environment=development`**, siempre y cuando no especifiquen un tamaño mínimo inferior a **`1`** ni un tamaño máximo superior a **`10`**. Siempre que sea posible, utilice etiquetas que lo ayuden a controlar el acceso a los grupos de escalado automático de su cuenta.

JSON

```
{
   "Version":"2012-10-17",
   "Statement": [{
      "Effect": "Allow",
      "Action": [
          "autoscaling:CreateAutoScalingGroup",
          "autoscaling:UpdateAutoScalingGroup"
      ],
      "Resource": "*",
      "Condition": {
          "StringEquals": { "autoscaling:ResourceTag/environment":
 "development" },
          "NumericGreaterThanEqualsIfExists": { "autoscaling:MinSize": 1 },
          "NumericLessThanEqualsIfExists": { "autoscaling:MaxSize": 10 }
      }
   }]
}
```

Como alternativa, si no utiliza etiquetas para controlar el acceso a los grupos de Auto Scaling, puede utilizarlas ARNs para identificar los grupos de Auto Scaling a los que se aplica la política de IAM.

Un grupo de Auto Scaling tiene el siguiente ARN.

```
"Resource": "arn:aws:autoscaling:region:account-
id:autoScalingGroup:*:autoScalingGroupName/my-asg"
```

También puede especificar varios ARN incluyéndolos en una lista. Para obtener más información acerca de cómo especificar los ARN de los recursos de Amazon EC2 Auto Scaling en el elemento `Resource`, consulte Recursos de políticas de Amazon EC2 Auto Scaling.

Controlar qué claves de etiqueta y valores de etiqueta se pueden utilizar

También puede utilizar condiciones en las políticas de IAM para controlar las claves de etiqueta y los valores de etiqueta que se pueden aplicar a los grupos de escalado automático. A fin de conceder a los usuarios permisos para crear o etiquetar un grupo de escalado automático únicamente si proporcionan etiquetas específicas, utilice la clave de condición `aws:RequestTag`. Para permitir únicamente claves de etiqueta específicas, utilice la clave de condición `aws:TagKeys` con el modificador `ForAllValues`.

La siguiente política requiere que los solicitantes especifiquen una etiqueta con la clave **`environment`** en la solicitud. El valor **`"?*"`** exige que haya algún valor para la clave de etiqueta. Para utilizar un valor comodín, debe utilizar el operador de condición `StringLike`.

JSON

```
{
   "Version":"2012-10-17",
   "Statement": [{
      "Effect": "Allow",
      "Action": [
          "autoscaling:CreateAutoScalingGroup",
          "autoscaling:CreateOrUpdateTags"
      ],
      "Resource": "*",
      "Condition": {
          "StringLike": { "aws:RequestTag/environment": "?*" }
      }
   }]
```

```
}
```

La política siguiente especifica que los solicitantes solo pueden etiquetar grupos de escalado automático con las etiquetas **purpose=webserver** y **cost-center=cc123**, y permite solo las etiquetas **purpose** y **cost-center** (no se pueden especificar otras etiquetas).

JSON

```
{
   "Version":"2012-10-17",
   "Statement": [{
      "Effect": "Allow",
      "Action": [
          "autoscaling:CreateAutoScalingGroup",
          "autoscaling:CreateOrUpdateTags"
      ],
      "Resource": "*",
      "Condition": {
          "StringEquals": {
              "aws:RequestTag/purpose": "webserver",
              "aws:RequestTag/cost-center": "cc123"
          },
          "ForAllValues:StringEquals": { "aws:TagKeys": ["purpose", "cost-
center"] }
      }
   }]
}
```

La siguiente política requiere que los solicitantes especifiquen al menos una etiqueta en la solicitud y permite únicamente las claves **cost-center** y **owner**.

JSON

```
{
   "Version":"2012-10-17",
   "Statement": [{
      "Effect": "Allow",
      "Action": [
          "autoscaling:CreateAutoScalingGroup",
```

```
            "autoscaling:CreateOrUpdateTags"
        ],
        "Resource": "*",
        "Condition": {
            "ForAnyValue:StringEquals": { "aws:TagKeys": ["cost-center", "owner"] }
        }
    }]
}
```

> **Note**
>
> En cuanto a las condiciones, la clave de condición no distingue entre mayúsculas y minúsculas, mientras que el valor de condición sí. Por lo tanto, para aplicar la distinción entre mayúsculas y minúsculas de una clave de etiqueta, utilice la clave de condición `aws:TagKeys`, donde la clave de etiqueta se especifica como valor en la condición.

Control de los grupos de escalado automático que se pueden eliminar

La siguiente política permite eliminar un grupo de escalado automático solo si el grupo tiene la etiqueta **`environment=development`**.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Action": "autoscaling:DeleteAutoScalingGroup",
        "Resource": "*",
        "Condition": {
            "StringEquals": { "aws:ResourceTag/environment": "development" }
        }
    }]
}
```

La siguiente política utiliza la clave de `autoscaling:ForceDelete` condición para controlar el acceso a la acción de la `DeleteAutoScalingGroup` API. Esto puede impedir que algunos usuarios

usen la operación de borrado forzado, que termina todas las instancias dentro de un grupo de Auto Scaling al eliminar el grupo.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [{
        "Effect": "Deny",
        "Action": "autoscaling:DeleteAutoScalingGroup",
        "Resource": "*",
        "Condition": {
            "Bool": {
                "autoscaling:ForceDelete": "true"
            }
        }
    }]
}
```

Como alternativa, si no utiliza claves de condición para controlar el acceso a los grupos de Auto Scaling, puede especificar los recursos ARNs del `Resource` elemento para controlar el acceso.

La siguiente política otorga a los usuarios permisos para usar la acción de `DeleteAutoScalingGroup` API, pero solo para los grupos de Auto Scaling cuyo nombre comience por**`devteam-`**.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "autoscaling:DeleteAutoScalingGroup",
            "Resource": "arn:aws:autoscaling:us-
east-1:111122223333:autoScalingGroup:*:autoScalingGroupName/devteam-*"
        }
    ]
}
```

También puede especificar varios ARNs incluyéndolos en una lista. Al incluir el UUID, se garantiza que el acceso se conceda al grupo de escalado automático especificado. El UUID de un grupo nuevo es distinto del UUID de un grupo eliminado que tenía el mismo nombre.

```
"Resource": [
    "arn:aws:autoscaling:region:account-
id:autoScalingGroup:uuid:autoScalingGroupName/devteam-1",
    "arn:aws:autoscaling:region:account-
id:autoScalingGroup:uuid:autoScalingGroupName/devteam-2",
    "arn:aws:autoscaling:region:account-
id:autoScalingGroup:uuid:autoScalingGroupName/devteam-3"
]
```

Controlar qué políticas de escalado se pueden eliminar

La siguiente política permite la acción `DeletePolicy` para eliminar una política de escalado. Sin embargo, también deniega la acción si el grupo de escalado automático sobre el que se actúa tiene la etiqueta **`environment=production`**. Siempre que sea posible, utilice etiquetas que lo ayuden a controlar el acceso a los grupos de escalado automático de su cuenta.

JSON

```
{
   "Version":"2012-10-17",
   "Statement": [{
      "Effect": "Allow",
      "Action": "autoscaling:DeletePolicy",
      "Resource": "*"
   },
   {
      "Effect": "Deny",
      "Action": "autoscaling:DeletePolicy",
      "Resource": "*",
      "Condition": {
          "StringEquals": { "autoscaling:ResourceTag/environment": "production" }
      }
   }]
}
```

Control del acceso a las acciones de actualización de instancias

La política siguiente concede permisos para iniciar, revertir y cancelar una actualización de instancias solo si el grupo de escalado automático sobre el que se actúa tiene la etiqueta **`environment=testing`**. Como las acciones `Describe` no admiten permisos de nivel de recursos, debe especificarlas en una instrucción aparte sin condiciones.

JSON

```
{
   "Version":"2012-10-17",
   "Statement": [{
      "Effect": "Allow",
      "Action": [
          "autoscaling:StartInstanceRefresh",
          "autoscaling:CancelInstanceRefresh",
          "autoscaling:RollbackInstanceRefresh"
      ],
      "Resource": "*",
      "Condition": {
          "StringEquals": { "autoscaling:ResourceTag/environment": "testing" }
      }
   },
   {
      "Effect": "Allow",
      "Action": "autoscaling:DescribeInstanceRefreshes",
      "Resource": "*"
   }]
}
```

Para especificar la configuración deseada en la llamada `StartInstanceRefresh`, es posible que los usuarios necesiten algunos permisos relacionados, tales como:

- ec2: RunInstances — Para lanzar instancias de EC2 mediante una plantilla de lanzamiento, el usuario debe tener el `ec2:RunInstances` permiso establecido en una política de IAM. Para obtener más información, consulte Control del uso de la plantilla de inicialización de Amazon EC2 en los grupos de escalado automático.
- ec2: CreateTags — Para lanzar instancias de EC2 a partir de una plantilla de lanzamiento que añada etiquetas a las instancias y los volúmenes al crearlas, el usuario debe tener el

`ec2:CreateTags` permiso establecido en una política de IAM. Para obtener más información, consulte Permisos necesarios para etiquetar instancias y volúmenes.

- iam: PassRole — Para lanzar instancias de EC2 desde una plantilla de lanzamiento que contenga un perfil de instancia (un contenedor para un rol de IAM), el usuario también debe tener el `iam:PassRole` permiso establecido en una política de IAM. Para obtener más información y una política de IAM de ejemplo, consulte Rol de IAM para aplicaciones que se ejecuten en instancias de Amazon EC2.
- ssm: GetParameters — Para lanzar instancias de EC2 a partir de una plantilla de lanzamiento que utilice un AWS Systems Manager parámetro, el usuario también debe tener el permiso establecido en una política de IAM. `ssm:GetParameters` Para obtener más información, consulte Uso AWS Systems Manager parámetros en lugar de ID de AMI en las plantillas de lanzamiento.

Creación de un rol vinculado al servicio

Amazon EC2 Auto Scaling requiere permisos para crear un rol vinculado a un servicio la primera vez que un usuario Cuenta de AWS suyo invoque las acciones de la API Amazon EC2 Auto Scaling. Si el rol vinculado a servicios aún no existe, Amazon EC2 Auto Scaling lo crea en su cuenta. La función vinculada al servicio otorga permisos a Amazon EC2 Auto Scaling para que pueda llamar a otras Servicios de AWS personas en su nombre.

Para que la creación automática de roles se realice correctamente, los usuarios deben disponer de permisos para la acción `iam:CreateServiceLinkedRole`.

```
"Action": "iam:CreateServiceLinkedRole"
```

A continuación, se muestra un ejemplo de una política de permisos que permite a un usuario crear un rol vinculado a servicios de Amazon EC2 Auto Scaling para Amazon EC2 Auto Scaling.

JSON

```
{
   "Version":"2012-10-17",
   "Statement": [{
      "Effect": "Allow",
      "Action": "iam:CreateServiceLinkedRole",
      "Resource": "arn:aws:iam::*:role/aws-service-role/
autoscaling.amazonaws.com/AWSServiceRoleForAutoScaling",
```

```
        "Condition": {
            "StringLike": { "iam:AWSServiceName":"autoscaling.amazonaws.com" }
        }
    }]
}
```

Controle qué función vinculada al servicio puede transferirse (mediante) PassRole

Los usuarios que creen o actualicen grupos de escalado automático y especifiquen un rol de sufijo personalizado vinculado al servicio en la solicitud necesitan el permiso `iam:PassRole`.

Puedes usar el `iam:PassRole` permiso para proteger la seguridad de las claves administradas por tus AWS KMS clientes si permites que diferentes roles vinculados al servicio accedan a diferentes claves. En función de las necesidades de su organización, es posible que tenga una clave para el equipo de desarrollo, otra para el equipo de control de calidad y otra para el equipo de finanzas. En primer lugar, cree un rol vinculado al servicio que tenga acceso a la clave necesaria, por ejemplo, un rol vinculado a servicios llamado AWSServiceRoleForAutoScaling_devteamkeyaccess. A continuación, conecte la política a una identidad de IAM, como un usuario o un rol.

La siguiente política concede permisos a los usuarios para pasar el rol **`AWSServiceRoleForAutoScaling_devteamkeyaccess`** a cualquier grupo de escalado automático cuyo nombre comience con **`devteam-`**. Si la identidad de IAM que crea el grupo de escalado automático intenta especificar un rol vinculado a un servicio diferente, recibirá un error. Si eligen la opción de no especificar un rol vinculado a servicios, el rol AWSServiceRoleForAutoScaling predeterminado se utiliza en su lugar.

Para obtener más información acerca de los roles vinculados a servicios con un sufijo personalizado, consulte Roles vinculados a servicios de Amazon EC2 Auto Scaling.

Prevención de la sustitución confusa entre servicios

El problema de la sustitución confusa es un problema de seguridad en el que una entidad que no tiene permiso para realizar una acción puede obligar a una entidad con más privilegios a realizar la acción.

En este caso AWS, la suplantación de identidad entre servicios puede provocar el confuso problema de un diputado. La suplantación entre servicios puede producirse cuando un servicio (el servicio que lleva a cabo las llamadas) llama a otro servicio (el servicio al que se llama). El servicio que lleva a

cabo las llamadas se puedes manipular para utilizar sus permisos a fin de actuar en función de los recursos de otro cliente de una manera en la que no debe tener permiso para acceder.

Para evitarlo, AWS proporciona herramientas que le ayudan a proteger los datos de todos los servicios cuyos directores de servicio tengan acceso a los recursos de su cuenta. Recomendamos utilizar las claves de contexto de condición globales `aws:SourceArn` y `aws:SourceAccount` en las políticas de confianza para roles de servicio de Amazon EC2 Auto Scaling. Estas claves limitan los permisos que Amazon EC2 Auto Scaling otorga a otro servicio para el recurso.

Los valores de los `SourceAccount` campos `SourceArn` y se establecen cuando Amazon EC2 Auto Scaling AWS Security Token Service usa AWS STS() para asumir una función en su nombre.

Para utilizar las claves de condición globales `aws:SourceArn` o `aws:SourceAccount`, establezca el valor en el nombre de recurso de Amazon (ARN) o en la cuenta del recurso que almacena Amazon EC2 Auto Scaling. Siempre que sea posible, utilice `aws:SourceArn`, que es más específico. Establezca el valor en el ARN o en un patrón de ARN con caracteres comodín (`*`) para las partes desconocidas del ARN. Si no conoce el ARN del recurso, utilice `aws:SourceAccount` en su lugar.

En el ejemplo siguiente, se muestra cómo se pueden utilizar las claves de contexto de condición globales `aws:SourceArn` y `aws:SourceAccount` en Amazon EC2 Auto Scaling para evitar el problema del suplente confuso.

Ejemplo: uso de las claves de condición **`aws:SourceArn`** y **`aws:SourceAccount`**

Un rol que asume un servicio para realizar acciones en su nombre se denomina rol de servicio. En los casos en los que desee crear enlaces de ciclo de vida que envíen notificaciones a cualquier lugar que no sea Amazon EventBridge, debe crear un rol de servicio que permita a Amazon EC2 Auto Scaling enviar notificaciones a un tema de Amazon SNS o a una cola de Amazon SQS en su nombre. Si quiere que solo se asocie un grupo de escalado automático al acceso entre servicios, puede especificar la política de confianza del rol de servicio de la siguiente manera.

En este ejemplo de política de confianza se utilizan declaraciones de condición para limitar la capacidad de `AssumeRole` en el rol de servicio a solo las acciones que afectan al grupo de escalado automático indicado en la cuenta especificada. Las condiciones `aws:SourceArn` y `aws:SourceAccount` se evalúan de forma independiente. Cualquier solicitud para usar el rol de servicio debe cumplir ambas condiciones.

Antes de utilizar esta política, reemplace la región, el ID de la cuenta, el UUID y el nombre del grupo por valores válidos de su cuenta.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Sid": "ConfusedDeputyPreventionExamplePolicy",
        "Effect": "Allow",
        "Principal": {
            "Service": "autoscaling.amazonaws.com"
        },
        "Action": "sts:AssumeRole",
        "Condition": {
            "ArnLike": {
                "aws:SourceArn": "arn:aws:autoscaling:us-
east-1:111122223333:autoScalingGroup:uuid:autoScalingGroupName/my-asg"
            },
            "StringEquals": {
                "aws:SourceAccount": "111122223333"
            }
        }
    }
}
```

En el ejemplo anterior:

- El elemento `Principal` especifica la entidad principal del servicio (`autoscaling.amazonaws.com`).
- El elemento `Action` especifica la acción `sts:AssumeRole`.
- El elemento `Condition` especifica las claves de condición globales `aws:SourceArn` y `aws:SourceAccount`. El ARN del origen incluye el ID de cuenta, por lo que no es necesario utilizar `aws:SourceAccount` con `aws:SourceArn`.

Información adicional

Para obtener más información, consulte Claves de contexto de condición globales de AWS, Problema del suplente confuso y Actualización de una política de confianza de rol (consola) en la Guía del usuario de IAM.

Control del uso de la plantilla de inicialización de Amazon EC2 en los grupos de escalado automático

Amazon EC2 Auto Scaling admite el uso de plantillas de lanzamiento de Amazon EC2 con los grupos de Auto Scaling. Le recomendamos que permita a los usuarios crear grupos de Auto Scaling a partir de plantillas de lanzamiento, ya que de esta forma podrán utilizar las características más recientes de Amazon EC2 Auto Scaling y Amazon EC2. Por ejemplo, los usuarios deben especificar una plantilla de lanzamiento para utilizar una política de instancias mixtas.

Puede utilizar la política `AmazonEC2FullAccess` para proporcionar a los usuarios acceso completo para trabajar con recursos de Amazon EC2 Auto Scaling, plantillas de lanzamiento y otros recursos de EC2 en su cuenta. O bien puede crear sus propias políticas de IAM personalizadas para conceder a los usuarios permisos detallados que les permitan trabajar con plantillas de lanzamiento, tal y como se describe en este tema.

Una política de ejemplo que puede personalizar para su propio uso

A continuación, se incluye un ejemplo de una política de permisos básica que puede personalizar para su propio uso. La política proporciona a los usuarios permisos para crear, actualizar y eliminar todos los grupos de escalado automático, pero solo si el grupo utiliza la etiqueta **`purpose=testing`**. A continuación, concede permiso para todas las acciones `Describe`. Como las acciones `Describe` no admiten permisos de nivel de recursos, debe especificarlas en una instrucción aparte sin condiciones.

Las identidades de IAM (usuarios o roles) con esta política tienen permiso para crear o actualizar un grupo de escalado automático mediante una plantilla de lanzamiento porque también tienen permiso para usar la acción `ec2:RunInstances`.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
          {
            "Effect": "Allow",
            "Action": [
                "autoscaling:CreateAutoScalingGroup",
                "autoscaling:UpdateAutoScalingGroup",
                "autoscaling:DeleteAutoScalingGroup"
            ],
```

```
            "Resource": "*",
            "Condition": {
                "StringEquals": { "autoscaling:ResourceTag/purpose": "testing" }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "autoscaling:Describe*",
                "ec2:RunInstances"
            ],
            "Resource": "*"
        }
    ]
}
```

Es posible que los usuarios que crean o actualicen grupos de escalado automático necesiten algunos permisos relacionados, tales como:

- ec2: CreateTags — Para añadir etiquetas a las instancias y los volúmenes al crearlos, el usuario debe tener el `ec2:CreateTags` permiso establecido en una política de IAM. Para obtener más información, consulte Permisos necesarios para etiquetar instancias y volúmenes.
- iam: PassRole — Para lanzar instancias de EC2 a partir de una plantilla de lanzamiento que contenga un perfil de instancia (un contenedor para un rol de IAM), el usuario también debe tener el `iam:PassRole` permiso establecido en una política de IAM. Para obtener más información y una política de IAM de ejemplo, consulte Rol de IAM para aplicaciones que se ejecuten en instancias de Amazon EC2.
- ssm: GetParameters — Para lanzar instancias de EC2 a partir de una plantilla de lanzamiento que utilice un AWS Systems Manager parámetro, el usuario también debe tener el permiso establecido en una política de IAM. `ssm:GetParameters` Para obtener más información, consulte Uso AWS Systems Manager parámetros en lugar de ID de AMI en las plantillas de lanzamiento.

Estos permisos para que las acciones se completen al lanzar instancias se comprueban cuando el usuario interactúa con un grupo de escalado automático. Para obtener más información, consulte Validación de permisos para `ec2:RunInstances` y `iam:PassRole`.

Los siguientes ejemplos muestran instrucciones de política que puede utilizar para controlar el acceso que los usuarios de IAM tienen para usar plantillas de lanzamiento.

Temas

- Requerimiento de plantillas de lanzamiento que tengan una etiqueta específica
- Requerimiento de una plantilla de lanzamiento y un número de versión
- Requiere el uso de la versión 2 del servicio de metadatos de instancias (IMDSv2)
- Restricción del acceso a recursos de Amazon EC2
- Permisos necesarios para etiquetar instancias y volúmenes
- Permisos adicionales de la plantilla de lanzamiento
- Validación de permisos para ec2:RunInstances y iam:PassRole
- Recursos relacionados

Requerimiento de plantillas de lanzamiento que tengan una etiqueta específica

Al conceder `ec2:RunInstances` permisos, puede especificar que los usuarios solo puedan usar plantillas de lanzamiento con etiquetas específicas o específicas IDs para limitar los permisos al lanzar instancias con una plantilla de lanzamiento. También puede controlar la AMI y otros recursos a los que cualquier persona que utilice plantillas de lanzamiento puede hacer referencia y utilizar al lanzar instancias especificando los permisos adicionales a nivel de recursos para la llamada `RunInstances`.

En el siguiente ejemplo, se restringen los permisos para que la acción `ec2:RunInstances` lance plantillas que se encuentran en la región especificada y que tienen la etiqueta **`purpose=testing`**. También permite a los usuarios acceder a los recursos especificados en una plantilla de lanzamiento: tipos de instancias AMIs, volúmenes, pares de claves, interfaces de red y grupos de seguridad.

Para obtener más información sobre el uso de políticas basadas en etiquetas con plantillas de inicialización, consulte Controle el acceso a las plantillas de inicialización con permisos de IAM en la Guía del usuario de Amazon EC2.

Requerimiento de una plantilla de lanzamiento y un número de versión

También puede usar los permisos de IAM para exigir que se especifique una plantilla de lanzamiento y el número de versión de la plantilla de lanzamiento al crear o actualizar los grupos de escalado automático.

En el ejemplo siguiente, se permite a los usuarios crear y actualizar grupos de escalado automático solo si se especifican una plantilla de lanzamiento y el número de versión de la plantilla de

lanzamiento. Si los usuarios con esta política omiten el número de versión para especificar la versión de la plantilla de lanzamiento `$Latest` o `$Default`, o intentan usar una configuración de lanzamiento en su lugar, se producirá un error en la acción.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "autoscaling:CreateAutoScalingGroup",
                "autoscaling:UpdateAutoScalingGroup"
            ],
            "Resource": "*",
            "Condition": {
                "Bool": { "autoscaling:LaunchTemplateVersionSpecified": "true" }
            }
        },
        {
            "Effect": "Deny",
            "Action": [
                "autoscaling:CreateAutoScalingGroup",
                "autoscaling:UpdateAutoScalingGroup"
            ],
            "Resource": "*",
            "Condition": {
                "Null": { "autoscaling:LaunchConfigurationName": "false" }
            }
        }
    ]
}
```

Requiere el uso de la versión 2 del servicio de metadatos de instancias (IMDSv2)

Para mayor seguridad, puedes configurar los permisos de tus usuarios para que requieran el uso de una plantilla de lanzamiento que lo requiera IMDSv2. Para obtener más información, consulte Configuración del servicio de metadatos de instancia en la Guía del usuario de Amazon EC2.

En el siguiente ejemplo, se especifica que los usuarios no pueden ejecutar la `ec2:RunInstances` acción a menos que la instancia también esté habilitada para requerir el uso de IMDSv2 (lo que se indica mediante`"ec2:MetadataHttpTokens":"required"`).

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "RequireImdsV2",
            "Effect": "Deny",
            "Action": "ec2:RunInstances",
            "Resource": "arn:aws:ec2:*:*:instance/*",
            "Condition": {
                "StringNotEquals": { "ec2:MetadataHttpTokens": "required" }
            }
        }
    ]
}
```

> **Tip**
>
> Para forzar el lanzamiento de instancias de escalado automático de reemplazo que utilicen una nueva plantilla de lanzamiento o una nueva versión de una plantilla de lanzamiento con las opciones de metadatos de instancia configuradas, puede iniciar la actualización de instancias. Para obtener más información, consulte Actualización de las instancias de escalado automático.

Restricción del acceso a recursos de Amazon EC2

Los siguientes ejemplos muestran cómo usar los permisos a nivel de recursos y recursos etiquetados para restringir el acceso a recursos de Amazon EC2.

Ejemplo 1: Restringir los lanzamientos de instancias de Amazon EC2 a recursos y tipos de instancias específicos

En el siguiente ejemplo se controla la configuración de las instancias que un usuario puede lanzar al restringir el acceso a los recursos de Amazon EC2. Para especificar los permisos de nivel de recurso para los recursos especificados en una plantilla de lanzamiento, debe incluir los recursos en la instrucción de la acción `RunInstances`.

En este ejemplo, hay dos instrucciones:

- La primera instrucción requiere que los usuarios inicien instancias en una subred específica (**`subnet-1a2b3c4d`**), utilizando un grupo de seguridad específico (**`sg-903004f88example`**) y una AMI específica (**`ami-04d5cc9b88example`**). También permite a los usuarios acceder a los recursos especificados en una plantilla de lanzamiento: interfaces de red, pares de claves y volúmenes.
- La segunda instrucción permite a los usuarios lanzar instancias utilizando únicamente los tipos de instancia **`t2.micro`** y **`t2.small`**, que usted podría utilizar para controlar costos.

  Sin embargo, tenga en cuenta que actualmente no existe una forma eficaz de impedir por completo que los usuarios que tienen permiso para inicializar instancias con una plantilla de inicialización inicien otros tipos de instancias. Esto se debe a que un tipo de instancia especificado en una plantilla de inicialización se puede anular para usar tipos de instancia que se definen mediante la selección del tipo de instancia basada en atributos.

Para obtener una lista completa de los permisos de nivel de recurso que puede utilizar para controlar la configuración de las instancias que un usuario puede lanzar, consulte Acciones, recursos y claves de condición para Amazon EC2 en la Referencia de autorizaciones de servicio.

Ejemplo 2: Requerir etiquetas y restringir el acceso a los recursos para el lanzamiento de instancias de Amazon EC2

El siguiente ejemplo de política muestra cómo utilizar condiciones en la política basada en identidades para controlar el acceso a los recursos de Amazon EC2 basados en etiquetas.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "InstanceTags",
            "Effect": "Allow",
```

```
            "Action": [
                "ec2:RunInstances"
            ],
            "Resource": [
                "arn:aws:ec2:us-east-1:555555555555:instance/*"
            ],
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/owner": "dev"
                }
            }
        },
        {
            "Sid": "InstanceBoundaries",
            "Effect": "Allow",
            "Action": [
                "ec2:RunInstances"
            ],
            "Resource": [
                "arn:aws:ec2:us-east-1::image/*",
                "arn:aws:ec2:us-east-1:555555555555:subnet/*",
                "arn:aws:ec2:us-east-1:555555555555:network-interface/*"
            ]
        },
        {
            "Sid": "InstanceResourceTags",
            "Effect": "Allow",
            "Action": [
                "ec2:RunInstances"
            ],
            "Resource": [
                "arn:aws:ec2:us-east-1:555555555555:key-pair/*",
                "arn:aws:ec2:us-east-1:555555555555:security-group/*",
                "arn:aws:ec2:us-east-1:555555555555:launch-template/*",
                "arn:aws:ec2:us-east-1:555555555555:volume/*"
            ],
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/purpose": "testing"
                }
            }
        }
    ]
```

```
}
```

En este ejemplo, hay tres declaraciones:

- La primera declaración permite a los usuarios lanzar instancias en la región especificada solo si una etiqueta con `owner=dev` se utiliza en la solicitud.
- La segunda declaración otorga a los usuarios acceso a la AMI, la subred y la interfaz de red en la región especificada.
- La tercera declaración permite a los usuarios lanzar instancias en la región especificada con un par clave, un grupo de seguridad, una plantilla de lanzamiento y un volumen existentes que tengan la etiqueta `purpose=testing`.

Para obtener más información, consulte Controlar el acceso a AWS los recursos mediante etiquetas en la Guía del usuario de IAM.

Permisos necesarios para etiquetar instancias y volúmenes

En el siguiente ejemplo, se permite a los usuarios etiquetar instancias y volúmenes en el momento de la creación. Esta política es necesaria si hay etiquetas especificadas en la plantilla de lanzamiento. Para obtener más información, consulte Conceder permisos para etiquetar recursos durante la creación en la Guía del usuario de Amazon EC2.

Permisos adicionales de la plantilla de lanzamiento

Debe conceder permisos a los usuarios de la consola para las acciones `ec2:DescribeLaunchTemplates` y `ec2:DescribeLaunchTemplateVersions`. Sin estos permisos, los datos de la plantilla de lanzamiento no se pueden cargar en el asistente de grupos de Auto Scaling, y los usuarios no pueden completar el asistente para iniciar instancias con una plantilla de lanzamiento. Puede especificar estas acciones adicionales en el elemento `Action` de una instrucción de política de IAM.

Validación de permisos para **`ec2:RunInstances`** y **`iam:PassRole`**

Los usuarios pueden especificar qué versión de una plantilla de lanzamiento utiliza su grupo de escalado automático. Según sus permisos, puede ser una versión numerada específica o la versión `$Latest` o `$Default` de la plantilla de lanzamiento. Si es la última, tenga especial cuidado. Esto puede anular los permisos para `ec2:RunInstances` y `iam:PassRole` que pretendía restringir.

En esta sección se explica el escenario en el que se utiliza la versión más reciente o la versión predeterminada de la plantilla de lanzamiento con un grupo de escalado automático.

Cuando un usuario llama a las API de `CreateAutoScalingGroup`, `UpdateAutoScalingGroup`, o `StartInstanceRefresh`, Amazon EC2 Auto Scaling comprueba sus permisos con la versión de la plantilla de lanzamiento que sea la última o la versión predeterminada en ese momento antes de continuar con la solicitud. Esto valida los permisos de las acciones que se deben completar al lanzar instancias, como las acciones `ec2:RunInstances` y `iam:PassRole`. Para ello, emitimos una llamada de prueba a Amazon RunInstancesEC2 para validar si el usuario tiene los permisos necesarios para la acción, sin necesidad de realizar la solicitud. Cuando se devuelve una respuesta, la lee Amazon EC2 Auto Scaling. Si los permisos de usuario no permiten una acción determinada, Amazon EC2 Auto Scaling falla la solicitud y devuelve un error al usuario que contiene información sobre el permiso que falta.

Una vez finalizadas la verificación y la solicitud iniciales, cada vez que se lanzan instancias, Amazon EC2 Auto Scaling las lanza con la versión más reciente o predeterminada, incluso si ha cambiado, utilizando los permisos de su rol vinculado al servicio. Esto significa que un usuario que utiliza la plantilla de lanzamiento podría posiblemente actualizarla para transferir un rol de IAM a una instancia, incluso si no tiene el permiso `iam:PassRole`.

Utilice la clave de condición `autoscaling:LaunchTemplateVersionSpecified` si desea limitar quién tiene acceso a los grupos de configuración para usar la versión `$Latest` o `$Default`. Esto garantiza que el grupo Auto Scaling solo acepte una versión numerada específica cuando un usuario llame a la tecla `CreateAutoScalingGroup` y `UpdateAutoScalingGroup` APIs. Para ver un ejemplo que muestre cómo añadir esta clave de condición a una política de IAM, consulte Requerimiento de una plantilla de lanzamiento y un número de versión.

En el caso de los grupos de escalado automático que estén configurados para usar la versión de la plantilla de lanzamiento `$Latest` o `$Default`, considere limitar quién puede crear y gestionar versiones de la plantilla de lanzamiento, incluida la acción `ec2:ModifyLaunchTemplate` que permite al usuario especificar la versión predeterminada de la plantilla de lanzamiento. Para obtener más información, consulte Controle los permisos del control de versiones en la Guía del usuario de Amazon EC2.

Recursos relacionados

Para obtener más información sobre los permisos para ver, crear y eliminar plantillas de inicialización y versiones de plantillas de inicialización, consulte Controle el acceso a las plantillas de inicialización con permisos de IAM en la Guía del usuario de Amazon EC2.

Para obtener más información acerca de los permisos de nivel de recurso que puede utilizar para controlar el acceso a la llamada de `RunInstances`, consulte Acciones, recursos y claves de condición de Amazon EC2 en la Referencia de autorizaciones de servicio.

Rol de IAM para aplicaciones que se ejecuten en instancias de Amazon EC2

Las aplicaciones que se ejecutan en instancias de Amazon EC2 necesitan credenciales para acceder a otros Servicios de AWS. Para proporcionar estas credenciales de una forma segura, utilice un rol de IAM. El rol proporciona permisos temporales que la aplicación puede utilizar al acceder a otros recursos de AWS . Los permisos del rol determinan lo que puede hacer la aplicación.

Para las instancias de un grupo de escalado automático, debe crear una configuración o una plantilla de lanzamiento y elegir un perfil de instancias para asociarlo con las instancias. Un perfil de instancias es un contenedor de un rol de IAM que permite que Amazon EC2 transfiera el rol de IAM a una instancia cuando esta se lanza. En primer lugar, cree un rol de IAM que tenga todos los permisos necesarios para acceder a los AWS recursos. A continuación, cree el perfil de instancia y asígnele el rol.

> **Note**
>
> Como práctica recomendada, te recomendamos encarecidamente que crees el rol de forma que tenga los permisos mínimos Servicios de AWS que requiera tu aplicación para otros roles.

Contenido

- Requisitos previos
- Creación de una plantilla de lanzamiento
- Véase también

Requisitos previos

Cree el rol de IAM que la aplicación que se ejecuta en Amazon EC2 puede asumir. Elija los permisos adecuados, de modo que la aplicación a la que se le asigne posteriormente el rol pueda realizar las llamadas a la API específicas que necesita.

Si utiliza la consola de IAM en lugar de la AWS CLI o una de ellas AWS SDKs, la consola crea un perfil de instancia automáticamente y le asigna el mismo nombre que la función a la que corresponde.

Para crear un rol de IAM (consola)

1. Abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación de la izquierda, seleccione Roles.
3. Elija Crear rol.
4. En Select trusted entity (Seleccionar entidad de confianza), elija AWS service (Servicio de).
5. Para el caso de uso, elija EC2 y, luego, Next (Siguiente).
6. Si es posible, seleccione la política que desea utilizar para la política de permisos o elija Create policy (Crear política) para abrir una pestaña nueva del navegador y crear una política nueva desde cero. Para obtener más información, consulte Creación de políticas de IAM en la Guía del usuario de IAM. Después de crear la política, cierre esa pestaña y vuelva a la pestaña original. Seleccione la casilla situada junto a las políticas de permisos que desea conceder a los servicios.
7. (Opcional) Configure un límite de permisos. Se trata de una característica avanzada que está disponible para los roles de servicio. Para obtener más información, consulte Límites de permisos para las entidades de IAM en la Guía del usuario de IAM.
8. Elija Siguiente.
9. En la página Name, review, and create (Asignar nombre, revisar y crear), para Role name (Nombre del rol), escriba el nombre de un rol para ayudarle a identificar el propósito de este rol. El nombre debe ser único en su Cuenta de AWS. Como otros AWS recursos pueden hacer referencia al rol, no puedes editar el nombre del rol una vez creado.
10. Revise el rol y, a continuación, elija Crear rol.

Permisos de IAM

Use una política basada en identidades de IAM para controlar el acceso al nuevo rol de IAM. El permiso `iam:PassRole` es necesario para la identidad de IAM (usuario o rol) que crea o actualiza un grupo de escalado automático mediante una plantilla de lanzamiento que especifica un perfil de instancia.

La siguiente política de ejemplo otorga permisos para transferir únicamente los roles de IAM cuyo nombre comience por **`qateam-`**.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "arn:aws:iam::111122223333:role/qateam-*",
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": [
                        "ec2.amazonaws.com",
                        "ec2.amazonaws.com.cn"
                    ]
                }
            }
        }
    ]
}
```

> **Important**
>
> Para obtener información acerca de cómo Amazon EC2 Auto Scaling valida los permisos para la acción `iam:PassRole` de un grupo de escalado automático que utiliza una plantilla de lanzamiento, consulte Validación de permisos para `ec2:RunInstances` y `iam:PassRole`.

Creación de una plantilla de lanzamiento

Al crear la plantilla de lanzamiento mediante Consola de administración de AWS, en la sección de detalles avanzados, seleccione el rol en el perfil de la instancia de IAM. Para obtener más información, consulte Creación de una plantilla de lanzamiento mediante la configuración avanzada.

Al crear la plantilla de lanzamiento mediante el create-launch-templatecomando de AWS CLI, especifique el nombre del perfil de instancia de su función de IAM, como se muestra en el siguiente ejemplo.

```
aws ec2 create-launch-template --launch-template-name my-lt-with-instance-profile --
version-description version1 \
--launch-template-data
 '{"ImageId":"ami-04d5cc9b88example","InstanceType":"t2.micro","IamInstanceProfile":
{"Name":"my-instance-profile"}}'
```

Véase también

Para obtener más información de ayuda para comenzar a aprender y a utilizar roles de IAM para Amazon EC2, consulte:

- Roles de IAM para Amazon EC2 en la Guía del usuario de Amazon EC2
- Uso de perfiles de instancias y Uso de un rol de IAM para conceder permisos a aplicaciones que se ejecutan en instancias de Amazon EC2 en la Guía del usuario de IAM

Validación de la conformidad en Amazon EC2 Auto Scaling

Para saber si uno Servicio de AWS está dentro del ámbito de aplicación de programas de cumplimiento específicos, consulte Servicios de AWS Alcance por programa de cumplimiento Servicios de AWS de cumplimiento y elija el programa de cumplimiento que le interese. Para obtener información general, consulte Programas de AWS cumplimiento > Programas AWS .

Puede descargar informes de auditoría de terceros utilizando AWS Artifact. Para obtener más información, consulte Descarga de informes en AWS Artifact .

Su responsabilidad de cumplimiento al Servicios de AWS utilizarlos viene determinada por la confidencialidad de sus datos, los objetivos de cumplimiento de su empresa y las leyes y reglamentos aplicables. Para obtener más información sobre su responsabilidad de conformidad al utilizarlos Servicios de AWS, consulte AWS la documentación de seguridad.

Conformidad con DSS PCI

Amazon EC2 Auto Scaling admite el procesamiento, el almacenamiento y la transmisión de datos de tarjetas de crédito por parte de un comerciante o un proveedor de servicios. Se ha validado por estar conforme con la norma de seguridad de datos del sector de pagos con tarjeta (PCI DSS). Para obtener más información sobre PCI DSS, incluida la forma de solicitar una copia del PCI AWS Compliance Package, consulte PCI DSS Level 1.

Para obtener información sobre cómo lograr la conformidad con la PCI DSS para sus AWS cargas de trabajo, consulte la siguiente guía de conformidad:

- Estándar de seguridad de datos de la industria de tarjetas de pago (PCI DSS) 3.2.1 en adelante AWS

Amazon EC2 Auto Scaling y puntos de conexión de VPC de tipo interfaz

Para mejorar la posición de seguridad de su VPC, configure Amazon EC2 Auto Scaling para que utilice un punto de conexión de VPC de tipo interfaz. Los puntos de enlace de la interfaz utilizan AWS PrivateLink una tecnología que le permite acceder de forma privada a las API de Amazon EC2 Auto Scaling restringiendo todo el tráfico de red entre su VPC y Amazon EC2 Auto Scaling a la red. AWS Con los puntos de conexión de tipo interfaz, tampoco necesita una puerta de enlace de Internet, un dispositivo NAT ni una puerta de enlace privada virtual.

No es necesario que lo configure AWS PrivateLink, pero se recomienda hacerlo. Para obtener más información sobre AWS PrivateLink los puntos de enlace de VPC, consulte ¿Qué es? AWS PrivateLink en la Guía.AWS PrivateLink

Temas

- Creación de un punto de conexión de la VPC de tipo interfaz
- Creación de una política de puntos de conexión de VPC

Creación de un punto de conexión de la VPC de tipo interfaz

Cree un punto de enlace para Amazon EC2 Auto Scaling utilizando el siguiente nombre de servicio:

```
com.amazonaws.region.autoscaling
```

Para obtener más información, consulte Acceder a un AWS servicio mediante un punto final de VPC de interfaz en la AWS PrivateLink Guía.

No es necesario cambiar ninguna configuración de Amazon EC2 Auto Scaling. Amazon EC2 Auto Scaling llama a AWS otros servicios mediante puntos de enlace de servicio o puntos de enlace de VPC de interfaz privada, según se utilicen.

Creación de una política de puntos de conexión de VPC

Puede adjuntar una política a su punto de conexión de VPC para controlar el acceso a la API de Amazon EC2 Auto Scaling. La política especifica:

- La entidad de seguridad que puede realizar acciones.
- Las acciones que se pueden realizar.
- El recurso en el que se pueden realizar las acciones.

En el ejemplo siguiente, se muestra una política de puntos de conexión de VPC que deniega a todos los usuarios el permiso para eliminar una política de escalado a través del punto de enlace. La política de ejemplo también concede permiso a todos los usuarios para realizar todas las demás acciones.

```
{
   "Statement": [
        {
            "Action": "*",
            "Effect": "Allow",
            "Resource": "*",
            "Principal": "*"
        },
        {
            "Action": "autoscaling:DeleteScalingPolicy",
            "Effect": "Deny",
            "Resource": "*",
            "Principal": "*"
        }
    ]
}
```

Para obtener más información, consulte Uso de políticas de punto de conexión para controlar el acceso a puntos de conexión de VPC en la Guía de AWS PrivateLink .

Utilización de este servicio con un AWS SDK

AWS Los kits de desarrollo de software (SDK) están disponibles para muchos lenguajes de programación populares. Cada SDK proporciona una API, ejemplos de código y documentación que facilitan a los desarrolladores la creación de aplicaciones en su lenguaje preferido.

Documentación de SDK	Ejemplos de código
AWS SDK para C++	AWS SDK para C++ ejemplos de código
AWS CLI	AWS CLI ejemplos de código
AWS SDK para Go	AWS SDK para Go ejemplos de código
AWS SDK para Java	AWS SDK para Java ejemplos de código
AWS SDK para JavaScript	AWS SDK para JavaScript ejemplos de código
AWS SDK para Kotlin	AWS SDK para Kotlin ejemplos de código
AWS SDK para .NET	AWS SDK para .NET ejemplos de código
AWS SDK para PHP	AWS SDK para PHP ejemplos de código
Herramientas de AWS para PowerShell	Herramientas de AWS para PowerShell ejemplos de código
AWS SDK para Python (Boto3)	AWS SDK para Python (Boto3) ejemplos de código
AWS SDK para Ruby	AWS SDK para Ruby ejemplos de código
AWS SDK para Rust	AWS SDK para Rust ejemplos de código
AWS SDK para SAP ABAP	AWS SDK para SAP ABAP ejemplos de código
AWS SDK para Swift	AWS SDK para Swift ejemplos de código

Para obtener ejemplos específicos de este servicio, consulte Ejemplos de código para Auto Scaling usando AWS SDKs.

Ejemplo de disponibilidad

¿No encuentra lo que necesita? Solicite un ejemplo de código a través del enlace de Enviar comentarios que se encuentra al final de esta página.

Uso de las API de Amazon EC2 Auto Scaling

La API para Amazon EC2 Auto Scaling es la interfaz a nivel de protocolo para Amazon EC2 Auto Scaling. Al utilizar la API, debe formatear correctamente todas las solicitudes HTTPS y añadir una firma digital válida a cada solicitud. Para obtener más información, consulte la referencia de la API Amazon EC2 Auto Scaling.

La API Amazon EC2 Auto Scaling incluye las siguientes acciones de API:

Límites de la cuenta

- DescribeAccountLimits

Grupos de Auto Scaling

- AttachLoadBalancers
- AttachLoadBalancerTargetGroups
- CreateAutoScalingGroup
- DeleteAutoScalingGroup
- DescribeAutoScalingGroups
- DescribeLoadBalancers
- DescribeLoadBalancerTargetGroups
- DetachLoadBalancerTargetGroups
- DetachLoadBalancers
- UpdateAutoScalingGroup

Instancias de Auto Scaling

- AttachInstances
- DescribeAutoScalingInstances
- DetachInstances
- SetInstanceHealth
- SetInstanceProtection

- TerminateInstanceInAutoScalingGroup

Actualizaciones de instancias

- CancelInstanceRefresh
- DescribeInstanceRefreshes
- RollbackInstanceRefresh
- StartInstanceRefresh

Configuraciones de lanzamiento

- CreateLaunchConfiguration
- DeleteLaunchConfiguration
- DescribeLaunchConfigurations

Enlaces de ciclo de vida

- CompleteLifecycleAction
- DeleteLifecycleHook
- DescribeLifecycleHooks
- DescribeLifecycleHookTypes
- PutLifecycleHook
- RecordLifecycleActionHeartbeat

Supervisión

- DeleteNotificationConfiguration
- DescribeAutoScalingNotificationTypes
- DescribeMetricCollectionTypes
- DescribeNotificationConfigurations
- DisableMetricsCollection
- EnableMetricsCollection
- PutNotificationConfiguration

Escalado

- DeletePolicy
- DescribeAdjustmentTypes
- DescribePolicies
- DescribeScalingActivities
- DescribeScalingProcessTypes
- DescribeTerminationPolicyTypes
- ExecutePolicy
- PutScalingPolicy
- ResumeProcesses
- SetDesiredCapacity
- SuspendProcesses

Escalado programado

- BatchDeleteScheduledAction
- BatchPutScheduledUpdateGroupAction
- DeleteScheduledAction
- DescribeScheduledActions
- PutScheduledUpdateGroupAction

Estado de espera

- EnterStandby
- ExitStandby

Tags

- CreateOrUpdateTags
- DeleteTags
- DescribeTags

Piscinas cálidas

- PutWarmPool
- DescribeWarmPool
- DeleteWarmPool

Ejemplos de código para Auto Scaling usando AWS SDKs

Los siguientes ejemplos de código muestran cómo usar Auto Scaling con un kit de desarrollo de AWS software (SDK).

Los conceptos básicos son ejemplos de código que muestran cómo realizar las operaciones esenciales dentro de un servicio.

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

Los escenarios son ejemplos de código que muestran cómo llevar a cabo una tarea específica a través de llamadas a varias funciones dentro del servicio o combinado con otros Servicios de AWS.

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulteUtilización de este servicio con un AWS SDK. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Ejemplos de código

- Ejemplos básicos de Auto Scaling usando AWS SDKs
  - Introducción al escalado automático
  - Aprenda los conceptos básicos de Auto Scaling con un AWS SDK
  - Acciones para Auto Scaling usando AWS SDKs
    - Utilizar AttachInstances con una CLI
    - Úselo AttachLoadBalancerTargetGroups con un AWS SDK o CLI
    - Utilizar AttachLoadBalancers con una CLI
    - Utilizar CompleteLifecycleAction con una CLI
    - Úselo CreateAutoScalingGroup con un AWS SDK o CLI
    - Utilizar CreateLaunchConfiguration con una CLI
    - Utilizar CreateOrUpdateTags con una CLI
    - Úselo DeleteAutoScalingGroup con un AWS SDK o CLI
    - Utilizar DeleteLaunchConfiguration con una CLI
    - Utilizar DeleteLifecycleHook con una CLI
    - Utilizar DeleteNotificationConfiguration con una CLI

- Utilizar DeletePolicy con una CLI
- Utilizar DeleteScheduledAction con una CLI
- Utilizar DeleteTags con una CLI
- Utilizar DescribeAccountLimits con una CLI
- Utilizar DescribeAdjustmentTypes con una CLI
- Úselo DescribeAutoScalingGroups con un AWS SDK o CLI
- Úselo DescribeAutoScalingInstances con un AWS SDK o CLI
- Utilizar DescribeAutoScalingNotificationTypes con una CLI
- Utilizar DescribeLaunchConfigurations con una CLI
- Utilizar DescribeLifecycleHookTypes con una CLI
- Utilizar DescribeLifecycleHooks con una CLI
- Utilizar DescribeLoadBalancers con una CLI
- Utilizar DescribeMetricCollectionTypes con una CLI
- Utilizar DescribeNotificationConfigurations con una CLI
- Utilizar DescribePolicies con una CLI
- Úselo DescribeScalingActivities con un AWS SDK o CLI
- Utilizar DescribeScalingProcessTypes con una CLI
- Utilizar DescribeScheduledActions con una CLI
- Utilizar DescribeTags con una CLI
- Utilizar DescribeTerminationPolicyTypes con una CLI
- Utilizar DetachInstances con una CLI
- Utilizar DetachLoadBalancers con una CLI
- Úselo DisableMetricsCollection con un AWS SDK o CLI
- Úselo EnableMetricsCollection con un AWS SDK o CLI
- Utilizar EnterStandby con una CLI
- Utilizar ExecutePolicy con una CLI
- Utilizar ExitStandby con una CLI
- Utilizar PutLifecycleHook con una CLI
- Utilizar PutNotificationConfiguration con una CLI
- Utilizar PutScalingPolicy con una CLI

    - Utilizar PutScheduledUpdateGroupAction con una CLI
    - Utilizar RecordLifecycleActionHeartbeat con una CLI
    - Utilizar ResumeProcesses con una CLI
    - Úselo SetDesiredCapacity con un AWS SDK o CLI
    - Utilizar SetInstanceHealth con una CLI
    - Utilizar SetInstanceProtection con una CLI
    - Utilizar SuspendProcesses con una CLI
    - Úselo TerminateInstanceInAutoScalingGroup con un AWS SDK o CLI
    - Úselo UpdateAutoScalingGroup con un AWS SDK o CLI
- Escenarios para Auto Scaling usando AWS SDKs
    - Cree y gestione un servicio flexible mediante un AWS SDK

Ejemplos básicos de Auto Scaling usando AWS SDKs

Los siguientes ejemplos de código muestran cómo utilizar los conceptos básicos de Amazon EC2 Auto Scaling AWS SDKs con.

Ejemplos

- Introducción al escalado automático
- Aprenda los conceptos básicos de Auto Scaling con un AWS SDK
- Acciones para Auto Scaling usando AWS SDKs
    - Utilizar AttachInstances con una CLI
    - Úselo AttachLoadBalancerTargetGroups con un AWS SDK o CLI
    - Utilizar AttachLoadBalancers con una CLI
    - Utilizar CompleteLifecycleAction con una CLI
    - Úselo CreateAutoScalingGroup con un AWS SDK o CLI
    - Utilizar CreateLaunchConfiguration con una CLI
    - Utilizar CreateOrUpdateTags con una CLI
    - Úselo DeleteAutoScalingGroup con un AWS SDK o CLI
    - Utilizar DeleteLaunchConfiguration con una CLI
    - Utilizar DeleteLifecycleHook con una CLI

- Utilizar DeleteNotificationConfiguration con una CLI
- Utilizar DeletePolicy con una CLI
- Utilizar DeleteScheduledAction con una CLI
- Utilizar DeleteTags con una CLI
- Utilizar DescribeAccountLimits con una CLI
- Utilizar DescribeAdjustmentTypes con una CLI
- Úselo DescribeAutoScalingGroups con un AWS SDK o CLI
- Úselo DescribeAutoScalingInstances con un AWS SDK o CLI
- Utilizar DescribeAutoScalingNotificationTypes con una CLI
- Utilizar DescribeLaunchConfigurations con una CLI
- Utilizar DescribeLifecycleHookTypes con una CLI
- Utilizar DescribeLifecycleHooks con una CLI
- Utilizar DescribeLoadBalancers con una CLI
- Utilizar DescribeMetricCollectionTypes con una CLI
- Utilizar DescribeNotificationConfigurations con una CLI
- Utilizar DescribePolicies con una CLI
- Úselo DescribeScalingActivities con un AWS SDK o CLI
- Utilizar DescribeScalingProcessTypes con una CLI
- Utilizar DescribeScheduledActions con una CLI
- Utilizar DescribeTags con una CLI
- Utilizar DescribeTerminationPolicyTypes con una CLI
- Utilizar DetachInstances con una CLI
- Utilizar DetachLoadBalancers con una CLI
- Úselo DisableMetricsCollection con un AWS SDK o CLI
- Úselo EnableMetricsCollection con un AWS SDK o CLI
- Utilizar EnterStandby con una CLI
- Utilizar ExecutePolicy con una CLI
- Utilizar ExitStandby con una CLI
- Utilizar PutLifecycleHook con una CLI
- Utilizar PutNotificationConfiguration con una CLI

- Utilizar PutScalingPolicy con una CLI
- Utilizar PutScheduledUpdateGroupAction con una CLI
- Utilizar RecordLifecycleActionHeartbeat con una CLI
- Utilizar ResumeProcesses con una CLI
- Úselo SetDesiredCapacity con un AWS SDK o CLI
- Utilizar SetInstanceHealth con una CLI
- Utilizar SetInstanceProtection con una CLI
- Utilizar SuspendProcesses con una CLI
- Úselo TerminateInstanceInAutoScalingGroup con un AWS SDK o CLI
- Úselo UpdateAutoScalingGroup con un AWS SDK o CLI

Introducción al escalado automático

En los siguientes ejemplos de código se muestra cómo empezar a utilizar el escalado automático.

.NET

SDK para .NET (v4)

> **Note**
>
> Hay más información GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
namespace AutoScalingActions;

using Amazon.AutoScaling;

public class HelloAutoScaling
{
    /// <summary>
    /// Hello Amazon EC2 Auto Scaling. List EC2 Auto Scaling groups.
    /// </summary>
    /// <param name="args"></param>
    /// <returns>Async Task.</returns>
```

```
    static async Task Main(string[] args)
    {
        var client = new AmazonAutoScalingClient();

        Console.WriteLine("Welcome to Amazon EC2 Auto Scaling.");
        Console.WriteLine("Let's get a description of your Auto Scaling
 groups.");

        var response = await client.DescribeAutoScalingGroupsAsync();

        if (response.AutoScalingGroups == null ||
 response.AutoScalingGroups.Count == 0)
        {
            Console.WriteLine("Sorry, you don't have any Amazon EC2 Auto Scaling
 groups.");
            return;
        }
        response.AutoScalingGroups.ForEach(autoScalingGroup =>
        {

 Console.WriteLine($"{autoScalingGroup.AutoScalingGroupName}\t{autoScalingGroup.Availabil
        });

    }
}
```

- Para obtener más información sobre la API, consulta DescribeAutoScalingGroupsla Referencia AWS SDK para .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 CMake archivo CMake Lists.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 autoscaling)

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

# 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_autoscaling.cpp)

target_link_libraries(${PROJECT_NAME}
```

```
        ${AWSSDK_LINK_LIBRARIES})
```

Código del archivo de origen hello_autoscaling.cpp.

```
#include <aws/core/Aws.h>
#include <aws/autoscaling/AutoScalingClient.h>
#include <aws/autoscaling/model/DescribeAutoScalingGroupsRequest.h>
#include <iostream>

/*
 *  A "Hello Autoscaling" starter application which initializes an Amazon EC2
 Auto Scaling client and describes the
 *  Amazon EC2 Auto Scaling groups.
 *
 *  main function
 *
 *  Usage: 'hello_autoscaling'
 *
 */

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::AutoScaling::AutoScalingClient autoscalingClient(clientConfig);

        std::vector<Aws::String> groupNames;
        Aws::String nextToken; // Used for pagination.

        do {

            Aws::AutoScaling::Model::DescribeAutoScalingGroupsRequest request;
            if (!nextToken.empty()) {
                request.SetNextToken(nextToken);
            }
```

```
            Aws::AutoScaling::Model::DescribeAutoScalingGroupsOutcome outcome =
                    autoscalingClient.DescribeAutoScalingGroups(request);

            if (outcome.IsSuccess()) {
                const Aws::Vector<Aws::AutoScaling::Model::AutoScalingGroup>
 &autoScalingGroups =
                        outcome.GetResult().GetAutoScalingGroups();
                for (auto &group: autoScalingGroups) {
                    groupNames.push_back(group.GetAutoScalingGroupName());
                }
                nextToken = outcome.GetResult().GetNextToken();
            } else {
                std::cerr << "Error with AutoScaling::DescribeAutoScalingGroups.
 "
                          << outcome.GetError().GetMessage()
                          << std::endl;
                result = 1;
                break;
            }
        } while (!nextToken.empty());

        std::cout << "Found " << groupNames.size() << " AutoScaling groups." <<
 std::endl;
        for (auto &groupName: groupNames) {
            std::cout << "AutoScaling group: " << groupName << std::endl;
        }

    }


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

- Para obtener más información sobre la API, consulte DescribeAutoScalingGroupsla Referencia AWS SDK para C++ de la 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.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.autoscaling.AutoScalingClient;
import software.amazon.awssdk.services.autoscaling.model.AutoScalingGroup;
import
 software.amazon.awssdk.services.autoscaling.model.DescribeAutoScalingGroupsResponse;
import java.util.List;

/**
 * Before running this SDK for Java (v2) code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class DescribeAutoScalingGroups {
    public static void main(String[] args) throws InterruptedException {
        AutoScalingClient autoScalingClient = AutoScalingClient.builder()
                .region(Region.US_EAST_1)
                .build();

        describeGroups(autoScalingClient);
    }

    public static void describeGroups(AutoScalingClient autoScalingClient) {
        DescribeAutoScalingGroupsResponse response =
 autoScalingClient.describeAutoScalingGroups();
        List<AutoScalingGroup> groups = response.autoScalingGroups();
        groups.forEach(group -> {
            System.out.println("Group Name: " + group.autoScalingGroupName());
            System.out.println("Group ARN: " + group.autoScalingGroupARN());
```

```
        });
    }
}
```

- Para obtener más información sobre la API, consulta DescribeAutoScalingGroupsla Referencia AWS SDK for Java 2.x de la API.

PHP

SDK para PHP

> **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 function helloService()
    {
        $autoScalingClient = new AutoScalingClient([
            'region' => 'us-west-2',
            'version' => 'latest',
            'profile' => 'default',
        ]);

        $groups = $autoScalingClient->describeAutoScalingGroups([]);
        var_dump($groups);
    }
```

- Para obtener más información sobre la API, consulta DescribeAutoScalingGroupsla Referencia AWS SDK para PHP 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.

```
import boto3


def hello_autoscaling(autoscaling_client):
    """
    Use the AWS SDK for Python (Boto3) to create an Amazon EC2 Auto Scaling
 client and list
    some of the Auto Scaling groups in your account.
    This example uses the default settings specified in your shared credentials
    and config files.

    :param autoscaling_client: A Boto3 Amazon EC2 Auto Scaling client object.
    """
    print(
        "Hello, Amazon EC2 Auto Scaling! Let's list up to ten of you Auto Scaling
 groups:"
    )
    response = autoscaling_client.describe_auto_scaling_groups()
    groups = response.get("AutoScalingGroups", [])
    if groups:
        for group in groups:
            print(f"\t{group['AutoScalingGroupName']}:
 {group['AvailabilityZones']}")
    else:
        print("There are no Auto Scaling groups in your account.")


if __name__ == "__main__":
    hello_autoscaling(boto3.client("autoscaling"))
```

- Para obtener más información sobre la API, consulta DescribeAutoScalingGroupsla 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-autoscaling'
require 'logger'

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

  # Gets and prints a list of Auto Scaling groups for the account.
  def list_auto_scaling_groups
    paginator = @client.describe_auto_scaling_groups
    auto_scaling_groups = []
    paginator.each_page do |page|
      auto_scaling_groups.concat(page.auto_scaling_groups)
    end

    if auto_scaling_groups.empty?
      @logger.info('No Auto Scaling groups found for this account.')
    else
      auto_scaling_groups.each do |group|
        @logger.info("Auto Scaling group name: #{group.auto_scaling_group_name}")
        @logger.info("  Group ARN:             #{group.auto_scaling_group_arn}")
```

```
        @logger.info("  Min/max/desired:       #{group.min_size}/
#{group.max_size}/#{group.desired_capacity}")
        @logger.info("\n")
      end
    end
  end
end

if $PROGRAM_NAME == __FILE__
  autoscaling_client = Aws::AutoScaling::Client.new
  manager = AutoScalingManager.new(autoscaling_client)
  manager.list_auto_scaling_groups
end
```

- Para obtener más información sobre la API, consulta DescribeAutoScalingGroupsla Referencia AWS SDK para 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 list_groups(client: &Client) -> Result<(), Error> {
    let resp = client.describe_auto_scaling_groups().send().await?;

    println!("Groups:");

    let groups = resp.auto_scaling_groups();

    for group in groups {
        println!(
            "Name:  {}",
            group.auto_scaling_group_name().unwrap_or("Unknown")
        );
```

```
        println!(
            "Arn:   {}",
            group.auto_scaling_group_arn().unwrap_or("unknown"),
        );
        println!("Zones: {:?}", group.availability_zones(),);
        println!();
    }

    println!("Found {} group(s)", groups.len());

    Ok(())
}
```

- Para obtener más información sobre la API, consulta DescribeAutoScalingGroupsla 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, consulteUtilización de este servicio con un AWS SDK. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Aprenda los conceptos básicos de Auto Scaling con un AWS SDK

En el siguiente ejemplo de código, se muestra cómo:

- Crear un grupo de Amazon EC2 Auto Scaling con una plantilla de lanzamiento y zonas de disponibilidad y obtener información sobre las instancias en ejecución
- Habilita la recopilación de CloudWatch métricas de Amazon.
- Actualizar la capacidad deseada del grupo y esperar a que una instancia se inicie
- Terminar una instancia del grupo.
- Mostrar las actividades de escalado que se producen como respuesta a las solicitudes de los usuarios y a los cambios de capacidad
- Obtén estadísticas para CloudWatch las métricas y, a continuación, limpia los recursos.

.NET

SDK para .NET (v4)

> **Note**
>
> Hay más información GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
global using Amazon.AutoScaling;
global using Amazon.AutoScaling.Model;
global using Amazon.CloudWatch;
global using AutoScalingActions;
global using Microsoft.Extensions.DependencyInjection;
global using Microsoft.Extensions.Hosting;
global using Microsoft.Extensions.Logging;
global using Microsoft.Extensions.Logging.Console;
global using Microsoft.Extensions.Logging.Debug;



using Amazon.EC2;
using Microsoft.Extensions.Configuration;
using Host = Microsoft.Extensions.Hosting.Host;

namespace AutoScalingBasics;

public class AutoScalingBasics
{

    static async Task Main(string[] args)
    {
        // Set up dependency injection for Amazon EC2 Auto Scaling, Amazon
        // CloudWatch, and Amazon EC2.
        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<IAmazonAutoScaling>()
                .AddAWSService<IAmazonCloudWatch>()
                .AddAWSService<IAmazonEC2>()
                .AddTransient<AutoScalingWrapper>()
                .AddTransient<CloudWatchWrapper>()
                .AddTransient<EC2Wrapper>()
                .AddTransient<UIWrapper>()
            )
            .Build();


        var autoScalingWrapper =
host.Services.GetRequiredService<AutoScalingWrapper>();
        var cloudWatchWrapper =
host.Services.GetRequiredService<CloudWatchWrapper>();
        var ec2Wrapper = host.Services.GetRequiredService<EC2Wrapper>();
        var uiWrapper = host.Services.GetRequiredService<UIWrapper>();

        var configuration = new ConfigurationBuilder()
            .SetBasePath(Directory.GetCurrentDirectory())
            .AddJsonFile("settings.json") // Load test settings from .json file.
            .AddJsonFile("settings.local.json",
                true) // Optionally load local settings.
            .Build();

        var imageId = configuration["ImageId"];
        var instanceType = configuration["InstanceType"];
        var launchTemplateName = configuration["LaunchTemplateName"];

        launchTemplateName += Guid.NewGuid().ToString();

        // The name of the Auto Scaling group.
        var groupName = configuration["GroupName"];

        uiWrapper.DisplayTitle("Auto Scaling Basics");
        uiWrapper.DisplayAutoScalingBasicsDescription();

        // Create the launch template and save the template Id to use when
deleting the
        // launch template at the end of the application.
        var launchTemplateId = await
ec2Wrapper.CreateLaunchTemplateAsync(imageId!, instanceType!,
launchTemplateName);
```

```
        // Confirm that the template was created by asking for a description of
it.
        await ec2Wrapper.DescribeLaunchTemplateAsync(launchTemplateName);

        uiWrapper.PressEnter();

        var availabilityZones = await ec2Wrapper.ListAvailabilityZonesAsync();

        Console.WriteLine($"Creating an Auto Scaling group named {groupName}.");
        await autoScalingWrapper.CreateAutoScalingGroupAsync(
            groupName!,
            launchTemplateName,
            availabilityZones[0].ZoneName);

        // Keep checking the details of the new group until its lifecycle state
        // is "InService".
        Console.WriteLine($"Waiting for the Auto Scaling group to be active.");

        List<AutoScalingInstanceDetails> instanceDetails;

        do
        {
            instanceDetails = await
autoScalingWrapper.DescribeAutoScalingInstancesAsync(groupName!);
        }
        while (instanceDetails.Count <= 0);

        Console.WriteLine($"Auto scaling group {groupName} successfully
created.");
        Console.WriteLine($"{instanceDetails.Count} instances were created for
the group.");

        // Display the details of the Auto Scaling group.
        instanceDetails.ForEach(detail =>
        {
            Console.WriteLine($"Group name: {detail.AutoScalingGroupName}");
        });

        uiWrapper.PressEnter();

        uiWrapper.DisplayTitle("Metrics collection");
        Console.WriteLine($"Enable metrics collection for {groupName}");
        await autoScalingWrapper.EnableMetricsCollectionAsync(groupName!);
```

```
        // Show the metrics that are collected for the group.

        // Update the maximum size of the group to three instances.
        Console.WriteLine("--- Update the Auto Scaling group to increase max size
to 3 ---");
        int maxSize = 3;
        await autoScalingWrapper.UpdateAutoScalingGroupAsync(groupName!,
launchTemplateName, maxSize);

        Console.WriteLine("--- Describe all Auto Scaling groups to show the
current state of the group ---");
        var groups = await
autoScalingWrapper.DescribeAutoScalingGroupsAsync(groupName!);

        uiWrapper.DisplayGroupDetails(groups!);

        uiWrapper.PressEnter();

        uiWrapper.DisplayTitle("Describe account limits");
        await autoScalingWrapper.DescribeAccountLimitsAsync();

        uiWrapper.WaitABit(60, "Waiting for the resources to be ready.");

        uiWrapper.DisplayTitle("Set desired capacity");
        int desiredCapacity = 2;
        await autoScalingWrapper.SetDesiredCapacityAsync(groupName!,
desiredCapacity);

        Console.WriteLine("Get the two instance Id values");

        // Empty the group before getting the details again.
        groups.Clear();
        groups = await
autoScalingWrapper.DescribeAutoScalingGroupsAsync(groupName!);
        if (groups.Any())
        {
            foreach (AutoScalingGroup group in groups)
            {
                Console.WriteLine($"The group name is
{group.AutoScalingGroupName}");
                Console.WriteLine($"The group ARN is
{group.AutoScalingGroupARN}");
                var instances = group.Instances;
```

```
                foreach (Amazon.AutoScaling.Model.Instance instance in instances)
                {
                    Console.WriteLine($"The instance id is
{instance.InstanceId}");
                    Console.WriteLine($"The lifecycle state is
{instance.LifecycleState}");
                }
            }
        }

        uiWrapper.DisplayTitle("Scaling Activities");
        Console.WriteLine("Let's list the scaling activities that have occurred
for the group.");
        var activities = await
autoScalingWrapper.DescribeScalingActivitiesAsync(groupName!);
        if (activities.Any())
        {
            activities.ForEach(activity =>
            {
                Console.WriteLine($"The activity Id is {activity.ActivityId}");
                Console.WriteLine($"The activity details are
{activity.Details}");
            });
        }

        // Display the Amazon CloudWatch metrics that have been collected.
        var metrics = await
cloudWatchWrapper.GetCloudWatchMetricsAsync(groupName!);
        if (metrics.Any())
        {
            Console.WriteLine($"Metrics collected for {groupName}:");
            metrics.ForEach(metric =>
            {
                Console.Write($"Metric name: {metric.MetricName}\t");
                Console.WriteLine($"Namespace: {metric.Namespace}");
            });
        }

        var dataPoints = await
cloudWatchWrapper.GetMetricStatisticsAsync(groupName!);
        if (dataPoints.Any())
        {
            Console.WriteLine("Details for the metrics collected:");
            dataPoints.ForEach(detail => { Console.WriteLine(detail); });
```

```
        }

        // Disable metrics collection.
        Console.WriteLine("Disabling the collection of metrics for
{groupName}.");
        var success = await
autoScalingWrapper.DisableMetricsCollectionAsync(groupName!);

        if (success)
        {
            Console.WriteLine($"Successfully stopped metrics collection for
{groupName}.");
        }
        else
        {
            Console.WriteLine($"Could not stop metrics collection for
{groupName}.");
        }

        // Terminate all instances in the group.
        uiWrapper.DisplayTitle("Terminating Auto Scaling instances");
        Console.WriteLine("Now terminating all instances in the Auto Scaling
group.");

        if (groups is not null)
        {
            groups.ForEach(group =>
            {
                // Only delete instances in the AutoScaling group we created.
                if (group.AutoScalingGroupName == groupName)
                {
                    group.Instances.ForEach(async instance =>
                    {
                        await
autoScalingWrapper.TerminateInstanceInAutoScalingGroupAsync(instance.InstanceId);
                    });
                }
            });
        }

        // After all instances are terminated, delete the group.
        uiWrapper.DisplayTitle("Clean up resources");
        Console.WriteLine("Deleting the Auto Scaling group.");
        await autoScalingWrapper.DeleteAutoScalingGroupAsync(groupName!);
```

```
        // Delete the launch template.
        var deletedLaunchTemplateName = await
 ec2Wrapper.DeleteLaunchTemplateAsync(launchTemplateId);

        if (deletedLaunchTemplateName == launchTemplateName)
        {
            Console.WriteLine("Successfully deleted the launch template.");
        }

        Console.WriteLine("The demo is now concluded.");
    }
}


namespace AutoScalingBasics;

/// <summary>
/// A class to provide user interface methods for the EC2 AutoScaling Basics
/// scenario.
/// </summary>
public class UIWrapper
{
    public readonly string SepBar = new('-', Console.WindowWidth);

    /// <summary>
    /// Describe the steps in the EC2 AutoScaling Basics scenario.
    /// </summary>
    public void DisplayAutoScalingBasicsDescription()
    {
        Console.WriteLine("This code example performs the following
 operations:");
        Console.WriteLine(" 1. Creates an Amazon EC2 launch template.");
        Console.WriteLine(" 2. Creates an Auto Scaling group.");
        Console.WriteLine(" 3. Shows the details of the new Auto Scaling group");
        Console.WriteLine("    to show that only one instance was created.");
        Console.WriteLine(" 4. Enables metrics collection.");
        Console.WriteLine(" 5. Updates the Auto Scaling group to increase the");
        Console.WriteLine("    capacity to three.");
        Console.WriteLine(" 6. Describes Auto Scaling groups again to show the");
        Console.WriteLine("    current state of the group.");
        Console.WriteLine(" 7. Changes the desired capacity of the Auto
 Scaling");
        Console.WriteLine("    group to use an additional instance.");
```

```
        Console.WriteLine(" 8. Shows that there are now instances in the
 group.");
        Console.WriteLine(" 9. Lists the scaling activities that have occurred
 for the group.");
        Console.WriteLine("10. Displays the Amazon CloudWatch metrics that
 have");
        Console.WriteLine("    been collected.");
        Console.WriteLine("11. Disables metrics collection.");
        Console.WriteLine("12. Terminates all instances in the Auto Scaling
 group.");
        Console.WriteLine("13. Deletes the Auto Scaling group.");
        Console.WriteLine("14. Deletes the Amazon EC2 launch template.");
        PressEnter();
    }

    /// <summary>
    /// Display information about the Amazon Ec2 AutoScaling groups passed
    /// in the list of AutoScalingGroup objects.
    /// </summary>
    /// <param name="groups">A list of AutoScalingGroup objects.</param>
    public void DisplayGroupDetails(List<AutoScalingGroup> groups)
    {
        if (groups is null)
            return;

        groups.ForEach(group =>
        {
            Console.WriteLine($"Group name:\t{group.AutoScalingGroupName}");
            Console.WriteLine($"Group created:\t{group.CreatedTime}");
            Console.WriteLine($"Maximum number of instances:\t{group.MaxSize}");
            Console.WriteLine($"Desired number of instances:
\t{group.DesiredCapacity}");
        });
    }

    /// <summary>
    /// Display a message and wait until the user presses enter.
    /// </summary>
    public void PressEnter()
    {
        Console.Write("\nPress <Enter> to continue. ");
        _ = Console.ReadLine();
        Console.WriteLine();
    }
```

```
    /// <summary>
    /// Pad a string with spaces to center it on the console display.
    /// </summary>
    /// <param name="strToCenter">The string to be centered.</param>
    /// <returns>The padded string.</returns>
    public string CenterString(string strToCenter)
    {
        var padAmount = (Console.WindowWidth - strToCenter.Length) / 2;
        var leftPad = new string(' ', padAmount);
        return $"{leftPad}{strToCenter}";
    }

    /// <summary>
    /// Display a line of hyphens, the centered text of the title and another
    /// line of hyphens.
    /// </summary>
    /// <param name="strTitle">The string to be displayed.</param>
    public void DisplayTitle(string strTitle)
    {
        Console.WriteLine(SepBar);
        Console.WriteLine(CenterString(strTitle));
        Console.WriteLine(SepBar);
    }

    /// <summary>
    /// Display a countdown and wait for a number of seconds.
    /// </summary>
    /// <param name="numSeconds">The number of seconds to wait.</param>
    public void WaitABit(int numSeconds, string msg)
    {
        Console.WriteLine(msg);

        // Wait for the requested number of seconds.
        for (int i = numSeconds; i > 0; i--)
        {
            System.Threading.Thread.Sleep(1000);
            Console.Write($"{i}...");
        }

        PressEnter();
    }
}
```

Definir las funciones a las que llama el escenario para administrar plantillas de lanzamiento y métricas. Estas funciones incluyen Auto Scaling, Amazon EC2 y CloudWatch acciones.

```
namespace AutoScalingActions;

using Amazon.AutoScaling;
using Amazon.AutoScaling.Model;

/// <summary>
/// A class that includes methods to perform Amazon EC2 Auto Scaling
/// actions.
/// </summary>
public class AutoScalingWrapper
{
    private readonly IAmazonAutoScaling _amazonAutoScaling;

    /// <summary>
    /// Constructor for the AutoScalingWrapper class.
    /// </summary>
    /// <param name="amazonAutoScaling">The injected Amazon EC2 Auto Scaling
 client.</param>
    public AutoScalingWrapper(IAmazonAutoScaling amazonAutoScaling)
    {
        _amazonAutoScaling = amazonAutoScaling;
    }


    /// <summary>
    /// Create a new Amazon EC2 Auto Scaling group.
    /// </summary>
    /// <param name="groupName">The name to use for the new Auto Scaling
    /// group.</param>
    /// <param name="launchTemplateName">The name of the Amazon EC2 Auto Scaling
    /// launch template to use to create instances in the group.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> CreateAutoScalingGroupAsync(
        string groupName,
        string launchTemplateName,
        string availabilityZone)
    {
```

```
        var templateSpecification = new LaunchTemplateSpecification
        {
            LaunchTemplateName = launchTemplateName,
        };

        var zoneList = new List<string>
            {
                availabilityZone,
            };

        var request = new CreateAutoScalingGroupRequest
        {
            AutoScalingGroupName = groupName,
            AvailabilityZones = zoneList,
            LaunchTemplate = templateSpecification,
            MaxSize = 6,
            MinSize = 1
        };
        try
        {
            var response = await
_amazonAutoScaling.CreateAutoScalingGroupAsync(request);
            Console.WriteLine($"{groupName} Auto Scaling Group created");
            return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
        }
        catch (AlreadyExistsException)
        {
            Console.WriteLine($"{groupName} Auto Scaling Group already exists.");
            return true;
        }
    }



    /// <summary>
    /// Retrieve information about Amazon EC2 Auto Scaling quotas to the
    /// active AWS account.
    /// </summary>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DescribeAccountLimitsAsync()
    {
        var response = await _amazonAutoScaling.DescribeAccountLimitsAsync();
        Console.WriteLine("The maximum number of Auto Scaling groups is " +
response.MaxNumberOfAutoScalingGroups);
```

```
        Console.WriteLine("The current number of Auto Scaling groups is " +
 response.NumberOfAutoScalingGroups);
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }



    /// <summary>
    /// Retrieve a list of the Amazon EC2 Auto Scaling activities for an
    /// Amazon EC2 Auto Scaling group.
    /// </summary>
    /// <param name="groupName">The name of the Amazon EC2 Auto Scaling group.</
param>
    /// <returns>A list of Amazon EC2 Auto Scaling activities.</returns>
    public async Task<List<Activity>> DescribeScalingActivitiesAsync(
        string groupName)
    {
        var activities = new List<Activity>();
        var scalingActivitiesRequest = new DescribeScalingActivitiesRequest
        {
            AutoScalingGroupName = groupName,
            MaxRecords = 10,
        };

        var response = await
 _amazonAutoScaling.DescribeScalingActivitiesAsync(scalingActivitiesRequest);
        if (response.Activities != null)
        {
            activities = response.Activities;
        }
        return activities;
    }



    /// <summary>
    /// Get data about the instances in an Amazon EC2 Auto Scaling group.
    /// </summary>
    /// <param name="groupName">The name of the Amazon EC2 Auto Scaling group.</
param>
    /// <returns>A list of Amazon EC2 Auto Scaling details.</returns>
    public async Task<List<AutoScalingInstanceDetails>>
 DescribeAutoScalingInstancesAsync(
        string groupName)
```

```
    {
        var groups = await DescribeAutoScalingGroupsAsync(groupName);
        var instanceIds = new List<string>();
        var instanceDetails = new List<AutoScalingInstanceDetails>();
        if (groups != null)
        {
            groups.ForEach(group =>
            {
                if (group.AutoScalingGroupName == groupName && group.Instances !=
 null)
                {
                    group.Instances.ForEach(instance =>
                    {
                        instanceIds.Add(instance.InstanceId);
                    });
                }
            });

            var scalingGroupsRequest = new DescribeAutoScalingInstancesRequest
            {
                MaxRecords = 10,
                InstanceIds = instanceIds,
            };

            var response =
                await _amazonAutoScaling.DescribeAutoScalingInstancesAsync(
                    scalingGroupsRequest);
            if (response.AutoScalingInstances != null)
            {
                instanceDetails = response.AutoScalingInstances;
            }
        }

        return instanceDetails;
    }



    /// <summary>
    /// Retrieve a list of information about Amazon EC2 Auto Scaling groups.
    /// </summary>
    /// <param name="groupName">The name of the Amazon EC2 Auto Scaling group.</
param>
    /// <returns>A list of Amazon EC2 Auto Scaling groups.</returns>
```

```
    public async Task<List<AutoScalingGroup>> DescribeAutoScalingGroupsAsync(
        string groupName)
    {
        var groups = new List<AutoScalingGroup>();
        var groupList = new List<string>
            {
                groupName,
            };

        var request = new DescribeAutoScalingGroupsRequest
        {
            AutoScalingGroupNames = groupList,
        };

        var response = await
 _amazonAutoScaling.DescribeAutoScalingGroupsAsync(request);
        if (response.AutoScalingGroups != null)
        {
            groups = response.AutoScalingGroups;
        }

        return groups;
    }


    /// <summary>
    /// Delete an Auto Scaling group.
    /// </summary>
    /// <param name="groupName">The name of the Amazon EC2 Auto Scaling group.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteAutoScalingGroupAsync(
        string groupName)
    {
        var deleteAutoScalingGroupRequest = new DeleteAutoScalingGroupRequest
        {
            AutoScalingGroupName = groupName,
            ForceDelete = true,
        };

        var response = await
 _amazonAutoScaling.DeleteAutoScalingGroupAsync(deleteAutoScalingGroupRequest);
        if (response.HttpStatusCode == System.Net.HttpStatusCode.OK)
        {
```

```
            Console.WriteLine($"You successfully deleted {groupName}");
            return true;
        }

        Console.WriteLine($"Couldn't delete {groupName}.");
        return false;
    }


    /// <summary>
    /// Disable the collection of metric data for an Amazon EC2 Auto Scaling
    /// group.
    /// </summary>
    /// <param name="groupName">The name of the Auto Scaling group.</param>
    /// <returns>A Boolean value that indicates the success or failure of
    /// the operation.</returns>
    public async Task<bool> DisableMetricsCollectionAsync(string groupName)
    {
        var request = new DisableMetricsCollectionRequest
        {
            AutoScalingGroupName = groupName,
        };

        var response = await
_amazonAutoScaling.DisableMetricsCollectionAsync(request);
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Enable the collection of metric data for an Auto Scaling group.
    /// </summary>
    /// <param name="groupName">The name of the Auto Scaling group.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> EnableMetricsCollectionAsync(string groupName)
    {
        var listMetrics = new List<string>
            {
                "GroupMaxSize",
            };

        var collectionRequest = new EnableMetricsCollectionRequest
        {
            AutoScalingGroupName = groupName,
```

```
            Metrics = listMetrics,
            Granularity = "1Minute",
        };

        var response = await
 _amazonAutoScaling.EnableMetricsCollectionAsync(collectionRequest);
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Set the desired capacity of an Auto Scaling group.
    /// </summary>
    /// <param name="groupName">The name of the Auto Scaling group.</param>
    /// <param name="desiredCapacity">The desired capacity for the Auto
    /// Scaling group.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> SetDesiredCapacityAsync(
        string groupName,
        int desiredCapacity)
    {
        var capacityRequest = new SetDesiredCapacityRequest
        {
            AutoScalingGroupName = groupName,
            DesiredCapacity = desiredCapacity,
        };

        var response = await
 _amazonAutoScaling.SetDesiredCapacityAsync(capacityRequest);
        Console.WriteLine($"You have set the DesiredCapacity to
 {desiredCapacity}.");

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Terminate all instances in the Auto Scaling group in preparation for
    /// deleting the group.
    /// </summary>
    /// <param name="instanceId">The instance Id of the instance to terminate.</
param>
    /// <returns>A Boolean value that indicates the success or failure of
    /// the operation.</returns>
```

```
    public async Task<bool> TerminateInstanceInAutoScalingGroupAsync(
        string instanceId)
    {
        var request = new TerminateInstanceInAutoScalingGroupRequest
        {
            InstanceId = instanceId,
            ShouldDecrementDesiredCapacity = false,
        };

        var response = await
 _amazonAutoScaling.TerminateInstanceInAutoScalingGroupAsync(request);

        if (response.HttpStatusCode == System.Net.HttpStatusCode.OK)
        {
            Console.WriteLine($"You have terminated the instance: {instanceId}");
            return true;
        }

        Console.WriteLine($"Could not terminate {instanceId}");
        return false;
    }


    /// <summary>
    /// Update the capacity of an Auto Scaling group.
    /// </summary>
    /// <param name="groupName">The name of the Auto Scaling group.</param>
    /// <param name="launchTemplateName">The name of the EC2 launch template.</
param>
    /// <param name="maxSize">The maximum number of instances that can be
    /// created for the Auto Scaling group.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> UpdateAutoScalingGroupAsync(
        string groupName,
        string launchTemplateName,
        int maxSize)
    {
        var templateSpecification = new LaunchTemplateSpecification
        {
            LaunchTemplateName = launchTemplateName,
        };

        var groupRequest = new UpdateAutoScalingGroupRequest
        {
```

```
            MaxSize = maxSize,
            AutoScalingGroupName = groupName,
            LaunchTemplate = templateSpecification,
        };

        var response = await
 _amazonAutoScaling.UpdateAutoScalingGroupAsync(groupRequest);
        if (response.HttpStatusCode == System.Net.HttpStatusCode.OK)
        {
            Console.WriteLine($"You successfully updated the Auto Scaling group
 {groupName}.");
            return true;
        }
        else
        {
            return false;
        }
    }

}


namespace AutoScalingActions;

using Amazon.EC2;
using Amazon.EC2.Model;

public class EC2Wrapper
{
    private readonly IAmazonEC2 _amazonEc2;

    /// <summary>
    /// Constructor for the EC2Wrapper class.
    /// </summary>
    /// <param name="amazonEc2">The injected Amazon EC2 client.</param>
    public EC2Wrapper(IAmazonEC2 amazonEc2)
    {
        _amazonEc2 = amazonEc2;
    }

    /// <summary>
    /// Create a new Amazon EC2 launch template.
    /// </summary>
    /// <param name="imageId">The image Id to use for instances launched
```

```
    /// using the Amazon EC2 launch template.</param>
    /// <param name="instanceType">The type of EC2 instances to create.</param>
    /// <param name="launchTemplateName">The name of the launch template.</param>
    /// <returns>Returns the TemplateID of the new launch template.</returns>
    public async Task<string> CreateLaunchTemplateAsync(
        string imageId,
        string instanceType,
        string launchTemplateName)
    {
        var request = new CreateLaunchTemplateRequest
        {
            LaunchTemplateData = new RequestLaunchTemplateData
            {
                ImageId = imageId,
                InstanceType = instanceType,
            },
            LaunchTemplateName = launchTemplateName,
        };

        var response = await _amazonEc2.CreateLaunchTemplateAsync(request);

        return response.LaunchTemplate.LaunchTemplateId;
    }

    /// <summary>
    /// Delete an Amazon EC2 launch template.
    /// </summary>
    /// <param name="launchTemplateId">The TemplateId of the launch template to
    /// delete.</param>
    /// <returns>The name of the EC2 launch template that was deleted.</returns>
    public async Task<string> DeleteLaunchTemplateAsync(string launchTemplateId)
    {
        var request = new DeleteLaunchTemplateRequest
        {
            LaunchTemplateId = launchTemplateId,
        };

        var response = await _amazonEc2.DeleteLaunchTemplateAsync(request);
        return response.LaunchTemplate.LaunchTemplateName;
    }

    /// <summary>
    /// Retrieve information about an EC2 launch template.
    /// </summary>
```

```
    /// <param name="launchTemplateName">The name of the EC2 launch template.</
param>
    /// <returns>A Boolean value that indicates the success or failure of
    /// the operation.</returns>
    public async Task<bool> DescribeLaunchTemplateAsync(string
 launchTemplateName)
    {
        var request = new DescribeLaunchTemplatesRequest
        {
            LaunchTemplateNames = new List<string> { launchTemplateName, },
        };

        var response = await _amazonEc2.DescribeLaunchTemplatesAsync(request);

        if (response.LaunchTemplates is not null)
        {
            response.LaunchTemplates.ForEach(template =>
            {
                Console.Write($"{template.LaunchTemplateName}\t");
                Console.WriteLine(template.LaunchTemplateId);
            });

            return true;
        }

        return false;
    }

    /// <summary>
    /// Retrieve the availability zones for the current region.
    /// </summary>
    /// <returns>A collection of availability zones.</returns>
    public async Task<List<AvailabilityZone>> ListAvailabilityZonesAsync()
    {
        var response = await _amazonEc2.DescribeAvailabilityZonesAsync(
            new DescribeAvailabilityZonesRequest());

        return response.AvailabilityZones;
    }
}


namespace AutoScalingActions;
```

```
using Amazon.CloudWatch;
using Amazon.CloudWatch.Model;

/// <summary>
/// Contains methods to access Amazon CloudWatch metrics for the
/// Amazon EC2 Auto Scaling basics scenario.
/// </summary>
public class CloudWatchWrapper
{
    private readonly IAmazonCloudWatch _amazonCloudWatch;

    /// <summary>
    /// Constructor for the CloudWatchWrapper.
    /// </summary>
    /// <param name="amazonCloudWatch">The injected CloudWatch client.</param>
    public CloudWatchWrapper(IAmazonCloudWatch amazonCloudWatch)
    {
        _amazonCloudWatch = amazonCloudWatch;
    }

    /// <summary>
    /// Retrieve the metrics information collection for the Auto Scaling group.
    /// </summary>
    /// <param name="groupName">The name of the Auto Scaling group.</param>
    /// <returns>A list of Metrics collected for the Auto Scaling group.</
returns>
    public async Task<List<Metric>> GetCloudWatchMetricsAsync(string groupName)
    {
        var metrics = new List<Metric>();
        var filter = new DimensionFilter
        {
            Name = "AutoScalingGroupName",
            Value = $"{groupName}",
        };

        var request = new ListMetricsRequest
        {
            MetricName = "AutoScalingGroupName",
            Dimensions = new List<DimensionFilter> { filter },
            Namespace = "AWS/AutoScaling",
        };

        var response = await _amazonCloudWatch.ListMetricsAsync(request);
        if (response.Metrics != null)
```

```
        {
            metrics = response.Metrics;
        }
        return metrics;
    }

    /// <summary>
    /// Retrieve the metric data collected for an Amazon EC2 Auto Scaling group.
    /// </summary>
    /// <param name="groupName">The name of the Amazon EC2 Auto Scaling group.</
param>
    /// <returns>A list of data points.</returns>
    public async Task<List<Datapoint>> GetMetricStatisticsAsync(string groupName)
    {
        var dataPoints = new List<Datapoint>();
        var metricDimensions = new List<Dimension>
            {
                new Dimension
                {
                    Name = "AutoScalingGroupName",
                    Value = $"{groupName}",
                },
            };

        // The start time will be yesterday.
        var startTime = DateTime.UtcNow.AddDays(-1);

        var request = new GetMetricStatisticsRequest
        {
            MetricName = "AutoScalingGroupName",
            Dimensions = metricDimensions,
            Namespace = "AWS/AutoScaling",
            Period = 60, // 60 seconds.
            Statistics = new List<string>() { "Minimum" },
            StartTimeUtc = startTime,
            EndTimeUtc = DateTime.UtcNow,
        };

        var response = await _amazonCloudWatch.GetMetricStatisticsAsync(request);
        if (response.Datapoints != null)
        {
            dataPoints = response.Datapoints;
        }
```

```
        return dataPoints;
    }

}
```

- Para obtener detalles sobre la API, consulte los siguientes temas en la Referencia de la API de AWS SDK para .NET .
  - CreateAutoScalingGroup
  - DeleteAutoScalingGroup
  - DescribeAutoScalingGroups
  - DescribeAutoScalingInstances
  - DescribeScalingActivities
  - DisableMetricsCollection
  - EnableMetricsCollection
  - SetDesiredCapacity
  - TerminateInstanceInAutoScalingGroup
  - UpdateAutoScalingGroup

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.

```
//! Routine which demonstrates using an Auto Scaling group
//! to manage Amazon EC2 instances.
/*!
  \sa groupsAndInstancesScenario()
  \param clientConfig: AWS client configuration.
  \return bool: Successful completion.
 */
```

```
bool AwsDoc::AutoScaling::groupsAndInstancesScenario(
        const Aws::Client::ClientConfiguration &clientConfig) {
    Aws::String templateName;
    Aws::EC2::EC2Client ec2Client(clientConfig);

    std::cout << std::setfill('*') << std::setw(ASTERISK_FILL_WIDTH) << " "
              << std::endl;
    std::cout
            << "Welcome to the Amazon Elastic Compute Cloud (Amazon EC2) Auto
 Scaling "
            << "demo for managing groups and instances." << std::endl;
    std::cout << std::setfill('*') << std::setw(ASTERISK_FILL_WIDTH) << " \n"
              << std::endl;

    std::cout << "This example requires an EC2 launch template." << std::endl;
    if (askYesNoQuestion(
            "Would you like to use an existing EC2 launch template (y/n)?  ")) {

        // 1. Specify the name of an existing EC2 launch template.
        templateName = askQuestion(
                "Enter the name of the existing EC2 launch template.  ");

        Aws::EC2::Model::DescribeLaunchTemplatesRequest request;
        request.AddLaunchTemplateNames(templateName);
        Aws::EC2::Model::DescribeLaunchTemplatesOutcome outcome =
                ec2Client.DescribeLaunchTemplates(request);

        if (outcome.IsSuccess()) {
            std::cout << "Validated the EC2 launch template '" << templateName
                      << "' exists by calling DescribeLaunchTemplate." <<
 std::endl;
        }
        else {
            std::cerr << "Error validating the existence of the launch template.
 "
                      << outcome.GetError().GetMessage()
                      << std::endl;
        }
    }
    else { // 2.  Or create a new EC2 launch template.
        templateName = askQuestion("Enter the name for a new EC2 launch template:
 ");

        Aws::EC2::Model::CreateLaunchTemplateRequest request;
```

```
        request.SetLaunchTemplateName(templateName);

        Aws::EC2::Model::RequestLaunchTemplateData requestLaunchTemplateData;

requestLaunchTemplateData.SetInstanceType(EC2_LAUNCH_TEMPLATE_INSTANCE_TYPE);
        requestLaunchTemplateData.SetImageId(EC2_LAUNCH_TEMPLATE_IMAGE_ID);

        request.SetLaunchTemplateData(requestLaunchTemplateData);

        Aws::EC2::Model::CreateLaunchTemplateOutcome outcome =
                ec2Client.CreateLaunchTemplate(request);

        if (outcome.IsSuccess()) {
            std::cout << "The EC2 launch template '" << templateName << " was
created."
                      << std::endl;
        }
        else if (outcome.GetError().GetExceptionName() ==
                 "InvalidLaunchTemplateName.AlreadyExistsException") {
            std::cout << "The EC2 template '" << templateName << "' already
exists"
                      << std::endl;
        }
        else {
            std::cerr << "Error with EC2::CreateLaunchTemplate. "
                      << outcome.GetError().GetMessage()
                      << std::endl;
        }
    }
    Aws::AutoScaling::AutoScalingClient autoScalingClient(clientConfig);
    std::cout << "Let's create an Auto Scaling group." << std::endl;
    Aws::String groupName = askQuestion(
            "Enter a name for the Auto Scaling group:  ");
    // 3. Retrieve a list of EC2 Availability Zones.
    Aws::Vector<Aws::EC2::Model::AvailabilityZone> availabilityZones;
    {
        Aws::EC2::Model::DescribeAvailabilityZonesRequest request;

        Aws::EC2::Model::DescribeAvailabilityZonesOutcome outcome =
                ec2Client.DescribeAvailabilityZones(request);

        if (outcome.IsSuccess()) {
            std::cout
```

```
                    << "EC2 instances can be created in the following
Availability Zones:"
                    << std::endl;

            availabilityZones = outcome.GetResult().GetAvailabilityZones();
            for (size_t i = 0; i < availabilityZones.size(); ++i) {
                std::cout << "   " << i + 1 << ".  "
                          << availabilityZones[i].GetZoneName() << std::endl;
            }
        }
        else {
            std::cerr << "Error with EC2::DescribeAvailabilityZones. "
                      << outcome.GetError().GetMessage()
                      << std::endl;
            cleanupResources("", templateName, autoScalingClient, ec2Client);
            return false;
        }
    }

    int availabilityZoneChoice = askQuestionForIntRange(
            "Choose an Availability Zone:  ", 1,
            static_cast<int>(availabilityZones.size()));
    // 4. Create an Auto Scaling group with the specified Availability Zone.
    {
        Aws::AutoScaling::Model::CreateAutoScalingGroupRequest request;
        request.SetAutoScalingGroupName(groupName);
        Aws::Vector<Aws::String> availabilityGroupZones;
        availabilityGroupZones.push_back(
                availabilityZones[availabilityZoneChoice - 1].GetZoneName());
        request.SetAvailabilityZones(availabilityGroupZones);
        request.SetMaxSize(1);
        request.SetMinSize(1);

        Aws::AutoScaling::Model::LaunchTemplateSpecification
launchTemplateSpecification;
        launchTemplateSpecification.SetLaunchTemplateName(templateName);
        request.SetLaunchTemplate(launchTemplateSpecification);

        Aws::AutoScaling::Model::CreateAutoScalingGroupOutcome outcome =
                autoScalingClient.CreateAutoScalingGroup(request);

        if (outcome.IsSuccess()) {
            std::cout << "Created Auto Scaling group '" << groupName << "'..."
                      << std::endl;
```

```
        }
        else if (outcome.GetError().GetErrorType() ==
                 Aws::AutoScaling::AutoScalingErrors::ALREADY_EXISTS_FAULT) {
            std::cout << "Auto Scaling group '" << groupName << "' already
exists."
                      << std::endl;
        }
        else {
            std::cerr << "Error with AutoScaling::CreateAutoScalingGroup. "
                      << outcome.GetError().GetMessage()
                      << std::endl;
            cleanupResources("", templateName, autoScalingClient, ec2Client);
            return false;
        }
    }

    Aws::Vector<Aws::AutoScaling::Model::AutoScalingGroup> autoScalingGroups;
    if (AwsDoc::AutoScaling::describeGroup(groupName, autoScalingGroups,
                                           autoScalingClient)) {
        std::cout << "Here is the Auto Scaling group description." << std::endl;
        if (!autoScalingGroups.empty()) {
            logAutoScalingGroupInfo(autoScalingGroups);
        }
    }
    else {
        cleanupResources(groupName, templateName, autoScalingClient, ec2Client);
        return false;
    }

    std::cout
            << "Waiting for the EC2 instance in the Auto Scaling group to become
active..."
            << std::endl;
    if (!waitForInstances(groupName, autoScalingGroups, autoScalingClient)) {
        cleanupResources(groupName, templateName, autoScalingClient, ec2Client);
        return false;
    }

    bool enableMetrics = askYesNoQuestion(
            "Do you want to collect metrics about the A"
            "Auto Scaling group during this demo (y/n)?  ");
    // 7. Optionally enable metrics collection for the Auto Scaling group.
    if (enableMetrics) {
        Aws::AutoScaling::Model::EnableMetricsCollectionRequest request;
```

```
        request.SetAutoScalingGroupName(groupName);

        request.AddMetrics("GroupMinSize");
        request.AddMetrics("GroupMaxSize");
        request.AddMetrics("GroupDesiredCapacity");
        request.AddMetrics("GroupInServiceInstances");
        request.AddMetrics("GroupTotalInstances");
        request.SetGranularity("1Minute");

        Aws::AutoScaling::Model::EnableMetricsCollectionOutcome outcome =
                autoScalingClient.EnableMetricsCollection(request);
        if (outcome.IsSuccess()) {
            std::cout << "Auto Scaling metrics have been enabled."
                      << std::endl;
        }
        else {
            std::cerr << "Error with AutoScaling::EnableMetricsCollection. "
                      << outcome.GetError().GetMessage()
                      << std::endl;
            cleanupResources(groupName, templateName, autoScalingClient,
ec2Client);
            return false;
        }
    }

    std::cout << "Let's update the maximum number of EC2 instances in '" <<
groupName <<
              "' from 1 to 3." << std::endl;
    askQuestion("Press enter to continue:  ", alwaysTrueTest);
    // 8. Update the Auto Scaling group, setting a new maximum size.
    {
        Aws::AutoScaling::Model::UpdateAutoScalingGroupRequest request;
        request.SetAutoScalingGroupName(groupName);
        request.SetMaxSize(3);

        Aws::AutoScaling::Model::UpdateAutoScalingGroupOutcome outcome =
                autoScalingClient.UpdateAutoScalingGroup(request);

        if (!outcome.IsSuccess()) {
            std::cerr << "Error with AutoScaling::UpdateAutoScalingGroup. "
                      << outcome.GetError().GetMessage()
                      << std::endl;
            cleanupResources(groupName, templateName, autoScalingClient,
ec2Client);
```

```
            return false;
        }
    }

    if (AwsDoc::AutoScaling::describeGroup(groupName, autoScalingGroups,
                                           autoScalingClient)) {
        if (!autoScalingGroups.empty()) {
            const auto &instances = autoScalingGroups[0].GetInstances();
            std::cout
                    << "The group still has one running EC2 instance, but it can
have up to 3.\n"
                    << std::endl;
            logAutoScalingGroupInfo(autoScalingGroups);
        }
        else {
            std::cerr
                    << "No EC2 launch groups were retrieved from DescribeGroup
request."
                    << std::endl;
            cleanupResources(groupName, templateName, autoScalingClient,
ec2Client);
            return false;
        }
    }

    std::cout << "\n" << std::setfill('*') << std::setw(ASTERISK_FILL_WIDTH) <<
"\n"
              << std::endl;
    std::cout << "Let's update the desired capacity in '" << groupName <<
              "' from 1 to 2." << std::endl;
    askQuestion("Press enter to continue:  ", alwaysTrueTest);
    //  9. Update the Auto Scaling group, setting a new desired capacity.
    {
        Aws::AutoScaling::Model::SetDesiredCapacityRequest request;
        request.SetAutoScalingGroupName(groupName);
        request.SetDesiredCapacity(2);

        Aws::AutoScaling::Model::SetDesiredCapacityOutcome outcome =
                autoScalingClient.SetDesiredCapacity(request);

        if (!outcome.IsSuccess()) {
            std::cerr << "Error with AutoScaling::SetDesiredCapacityRequest. "
                      << outcome.GetError().GetMessage()
                      << std::endl;
```

```
            cleanupResources(groupName, templateName, autoScalingClient,
ec2Client);
            return false;
        }
    }

    if (AwsDoc::AutoScaling::describeGroup(groupName, autoScalingGroups,
                                           autoScalingClient)) {
        if (!autoScalingGroups.empty()) {
            std::cout
                    << "Here is the current state of the group." << std::endl;
            logAutoScalingGroupInfo(autoScalingGroups);
        }
        else {
            std::cerr
                    << "No EC2 launch groups were retrieved from DescribeGroup
request."
                    << std::endl;
            cleanupResources(groupName, templateName, autoScalingClient,
ec2Client);
            return false;
        }
    }

    std::cout << "Waiting for the new EC2 instance to start..." << std::endl;
    waitForInstances(groupName, autoScalingGroups, autoScalingClient);

    std::cout << "\n" << std::setfill('*') << std::setw(ASTERISK_FILL_WIDTH) <<
"\n"
              << std::endl;

    std::cout << "Let's terminate one of the EC2 instances in " << groupName <<
"."
              << std::endl;
    std::cout << "Because the desired capacity is 2, another EC2 instance will
start "
              << "to replace the terminated EC2 instance."
              << std::endl;
    std::cout << "The currently running EC2 instances are:" << std::endl;

    if (autoScalingGroups.empty()) {
        std::cerr << "Error describing groups. No groups returned." << std::endl;
        cleanupResources(groupName, templateName, autoScalingClient, ec2Client);
        return false;
```

```
   }

   int instanceNumber = 1;
   Aws::Vector<Aws::String> instanceIDs = instancesToInstanceIDs(
           autoScalingGroups[0].GetInstances());
   for (const Aws::String &instanceID: instanceIDs) {
       std::cout << "   " << instanceNumber << ". " << instanceID << std::endl;
       ++instanceNumber;
   }

   instanceNumber = askQuestionForIntRange("Which EC2 instance do you want to
stop? ",
                                           1,

static_cast<int>(instanceIDs.size()));

   // 10. Terminate an EC2 instance in the Auto Scaling group.
   {
       Aws::AutoScaling::Model::TerminateInstanceInAutoScalingGroupRequest
request;
       request.SetInstanceId(instanceIDs[instanceNumber - 1]);
       request.SetShouldDecrementDesiredCapacity(false);

       Aws::AutoScaling::Model::TerminateInstanceInAutoScalingGroupOutcome
outcome =
               autoScalingClient.TerminateInstanceInAutoScalingGroup(request);

       if (outcome.IsSuccess()) {
           std::cout << "Waiting for EC2 instance with ID '"
                     << instanceIDs[instanceNumber - 1] << "' to terminate..."
                     << std::endl;
       }
       else {
           std::cerr << "Error with
AutoScaling::TerminateInstanceInAutoScalingGroup. "
                     << outcome.GetError().GetMessage()
                     << std::endl;
           cleanupResources(groupName, templateName, autoScalingClient,
ec2Client);
           return false;
       }
   }

   waitForInstances(groupName, autoScalingGroups, autoScalingClient);
```

```
   std::cout << "\n" << std::setfill('*') << std::setw(ASTERISK_FILL_WIDTH) <<
"\n"
             << std::endl;
   std::cout << "Let's get a report of scaling activities for EC2 launch group
'"
             << groupName << "'."
             << std::endl;
   askQuestion("Press enter to continue:  ", alwaysTrueTest);
   // 11. Get a description of activities for the Auto Scaling group.
   {
       Aws::AutoScaling::Model::DescribeScalingActivitiesRequest request;
       request.SetAutoScalingGroupName(groupName);

       Aws::Vector<Aws::AutoScaling::Model::Activity> allActivities;
       Aws::String nextToken; // Used for pagination;
       do {
           if (!nextToken.empty()) {
               request.SetNextToken(nextToken);
           }
           Aws::AutoScaling::Model::DescribeScalingActivitiesOutcome outcome =
                   autoScalingClient.DescribeScalingActivities(request);

           if (outcome.IsSuccess()) {
               const Aws::Vector<Aws::AutoScaling::Model::Activity> &activities
=
                       outcome.GetResult().GetActivities();
               allActivities.insert(allActivities.end(), activities.begin(),
activities.end());
               nextToken  = outcome.GetResult().GetNextToken();
           }
           else {
               std::cerr << "Error with AutoScaling::DescribeScalingActivities.
"
                         << outcome.GetError().GetMessage()
                         << std::endl;
               cleanupResources(groupName, templateName, autoScalingClient,
ec2Client);
               return false;
           }
       } while (!nextToken.empty());

       std::cout << "Found " << allActivities.size() << " activities."
                 << std::endl;
```

```
        std::cout << "Activities are ordered with the most recent first."
                  << std::endl;
        for (const Aws::AutoScaling::Model::Activity &activity: allActivities) {
            std::cout << activity.GetDescription() << std::endl;
            std::cout << activity.GetDetails() << std::endl;
        }
    }

    if (enableMetrics) {
        if (!logAutoScalingMetrics(groupName, clientConfig)) {
            cleanupResources(groupName, templateName, autoScalingClient,
 ec2Client);
            return false;
        }
    }

    std::cout << "Let's  clean up." << std::endl;
    askQuestion("Press enter to continue:  ", alwaysTrueTest);

    // 13. Disable metrics collection if enabled.
    if (enableMetrics) {
        Aws::AutoScaling::Model::DisableMetricsCollectionRequest request;
        request.SetAutoScalingGroupName(groupName);

        Aws::AutoScaling::Model::DisableMetricsCollectionOutcome outcome =
                autoScalingClient.DisableMetricsCollection(request);

        if (outcome.IsSuccess()) {
            std::cout << "Metrics collection has been disabled." << std::endl;
        }
        else {
            std::cerr << "Error with AutoScaling::DisableMetricsCollection. "
                      << outcome.GetError().GetMessage()
                      << std::endl;
            cleanupResources(groupName, templateName, autoScalingClient,
 ec2Client);
            return false;
        }
    }

    return cleanupResources(groupName, templateName, autoScalingClient,
 ec2Client);
}
```

```
//! Routine which waits for EC2 instances in an Auto Scaling group to
//! complete startup or shutdown.
/*!
 \sa waitForInstances()
 \param groupName: An Auto Scaling group name.
 \param autoScalingGroups: Vector to receive 'AutoScalingGroup' records.
 \param client: 'AutoScalingClient' instance.
 \return bool: Successful completion.
 */
bool AwsDoc::AutoScaling::waitForInstances(const Aws::String &groupName,

 Aws::Vector<Aws::AutoScaling::Model::AutoScalingGroup> &autoScalingGroups,
                                            const
 Aws::AutoScaling::AutoScalingClient &client) {
    bool ready = false;
    const std::vector<Aws::String> READY_STATES = {"InService", "Terminated"};

    int count = 0;
    int desiredCapacity = 0;
    std::this_thread::sleep_for(std::chrono::seconds(4));
    while (!ready) {
        if (WAIT_FOR_INSTANCES_TIMEOUT < count) {
            std::cerr << "Wait for instance timed out." << std::endl;
            return false;
        }

        std::this_thread::sleep_for(std::chrono::seconds(1));
        ++count;
        if (!describeGroup(groupName, autoScalingGroups, client)) {
            return false;
        }
        Aws::Vector<Aws::String> instanceIDs;
        if (!autoScalingGroups.empty()) {
            instanceIDs =
 instancesToInstanceIDs(autoScalingGroups[0].GetInstances());
            desiredCapacity = autoScalingGroups[0].GetDesiredCapacity();
        }

        if (instanceIDs.empty()) {
            if (desiredCapacity == 0) {
                break;
            }
            else {
                if ((count % 5) == 0) {
```

```
                    std::cout << "No instance IDs returned for group." <<
std::endl;
                }

                continue;
            }
        }

        // 6.  Check lifecycle state of the instances using
DescribeAutoScalingInstances.
        Aws::AutoScaling::Model::DescribeAutoScalingInstancesRequest request;
        request.SetInstanceIds(instanceIDs);

        Aws::AutoScaling::Model::DescribeAutoScalingInstancesOutcome outcome =
                client.DescribeAutoScalingInstances(request);

        if (outcome.IsSuccess()) {
            const
Aws::Vector<Aws::AutoScaling::Model::AutoScalingInstanceDetails>
&instancesDetails =
                    outcome.GetResult().GetAutoScalingInstances();
            ready = instancesDetails.size() >= desiredCapacity;
            for (const Aws::AutoScaling::Model::AutoScalingInstanceDetails
&details: instancesDetails) {
                if (!stringInVector(details.GetLifecycleState(), READY_STATES)) {
                    ready = false;
                    break;
                }
            }
            // Log the status while waiting.
            if (((count % 5) == 1) || ready) {
                logInstancesLifecycleState(instancesDetails);
            }
        }
        else {
            std::cerr << "Error with AutoScaling::DescribeAutoScalingInstances. "
                      << outcome.GetError().GetMessage()
                      << std::endl;
            return false;
        }
    }

    if (!describeGroup(groupName, autoScalingGroups, client)) {
        return false;
```

```
    }

    return true;
}

//! Routine to cleanup resources created in 'groupsAndInstancesScenario'.
/*!
 \sa cleanupResources()
 \param groupName: Optional Auto Scaling group name.
 \param templateName: Optional EC2 launch template name.
 \param autoScalingClient: 'AutoScalingClient' instance.
 \param ec2Client: 'EC2Client' instance.
\return bool: Successful completion.
 */
bool AwsDoc::AutoScaling::cleanupResources(const Aws::String &groupName,
                                           const Aws::String &templateName,
                                           const
 Aws::AutoScaling::AutoScalingClient &autoScalingClient,
                                           const Aws::EC2::EC2Client &ec2Client)
 {
    bool result = true;

    // 14. Delete the Auto Scaling group.
    if (!groupName.empty() &&
        (askYesNoQuestion(
                Aws::String("Delete the Auto Scaling group '") + groupName +
                "'  (y/n)?"))) {
        {
            Aws::AutoScaling::Model::UpdateAutoScalingGroupRequest request;
            request.SetAutoScalingGroupName(groupName);
            request.SetMinSize(0);
            request.SetDesiredCapacity(0);

            Aws::AutoScaling::Model::UpdateAutoScalingGroupOutcome outcome =
                    autoScalingClient.UpdateAutoScalingGroup(request);

            if (outcome.IsSuccess()) {
                std::cout
                        << "The minimum size and desired capacity of the Auto
 Scaling group "
                        << "was set to zero before terminating the instances."
                        << std::endl;
            }
            else {
```

```
                std::cerr << "Error with AutoScaling::UpdateAutoScalingGroup. "
                          << outcome.GetError().GetMessage() << std::endl;
                result = false;
            }
        }

        Aws::Vector<Aws::AutoScaling::Model::AutoScalingGroup> autoScalingGroups;
        if (AwsDoc::AutoScaling::describeGroup(groupName, autoScalingGroups,
                                               autoScalingClient)) {
            if (!autoScalingGroups.empty()) {
                Aws::Vector<Aws::String> instanceIDs = instancesToInstanceIDs(
                        autoScalingGroups[0].GetInstances());
                for (const Aws::String &instanceID: instanceIDs) {

Aws::AutoScaling::Model::TerminateInstanceInAutoScalingGroupRequest request;
                    request.SetInstanceId(instanceID);
                    request.SetShouldDecrementDesiredCapacity(true);


Aws::AutoScaling::Model::TerminateInstanceInAutoScalingGroupOutcome outcome =

autoScalingClient.TerminateInstanceInAutoScalingGroup(
                                    request);

                    if (outcome.IsSuccess()) {
                        std::cout << "Initiating termination of EC2 instance '"
                                  << instanceID << "'." << std::endl;
                    }
                    else {
                        std::cerr
                                << "Error with
AutoScaling::TerminateInstanceInAutoScalingGroup. "
                                << outcome.GetError().GetMessage() << std::endl;
                        result = false;
                    }
                }
            }

            std::cout
                    << "Waiting for the EC2 instances to terminate before
deleting the "
                    << "Auto Scaling group..." << std::endl;
            waitForInstances(groupName, autoScalingGroups, autoScalingClient);
        }
```

```
        {
            Aws::AutoScaling::Model::DeleteAutoScalingGroupRequest request;
            request.SetAutoScalingGroupName(groupName);

            Aws::AutoScaling::Model::DeleteAutoScalingGroupOutcome outcome =
                    autoScalingClient.DeleteAutoScalingGroup(request);

            if (outcome.IsSuccess()) {
                std::cout << "Auto Scaling group '" << groupName << "' was
deleted."
                          << std::endl;
            }
            else {
                std::cerr << "Error with AutoScaling::DeleteAutoScalingGroup. "
                          << outcome.GetError().GetMessage()
                          << std::endl;
                result = false;
            }
        }
    }

    // 15. Delete the EC2 launch template.
    if (!templateName.empty() && (askYesNoQuestion(
            Aws::String("Delete the EC2 launch template '") + templateName +
            "' (y/n)?"))) {
        Aws::EC2::Model::DeleteLaunchTemplateRequest request;
        request.SetLaunchTemplateName(templateName);

        Aws::EC2::Model::DeleteLaunchTemplateOutcome outcome =
                ec2Client.DeleteLaunchTemplate(request);

        if (outcome.IsSuccess()) {
            std::cout << "EC2 launch template '" << templateName << "' was
deleted."
                      << std::endl;
        }
        else {
            std::cerr << "Error with EC2::DeleteLaunchTemplate. "
                      << outcome.GetError().GetMessage()
                      << std::endl;
            result = false;
        }
    }
```

```
    return result;
}

//! Routine which retrieves Auto Scaling group descriptions.
/*!
 \sa describeGroup()
 \param groupName: An Auto Scaling group name.
 \param autoScalingGroups: Vector to receive 'AutoScalingGroup' records.
 \param client: 'AutoScalingClient' instance.
 \return bool: Successful completion.
 */
bool AwsDoc::AutoScaling::describeGroup(const Aws::String &groupName,

 Aws::Vector<Aws::AutoScaling::Model::AutoScalingGroup> &autoScalingGroup,
                                         const Aws::AutoScaling::AutoScalingClient
 &client) {
    // 5. Retrieve a description of the Auto Scaling group.
    Aws::AutoScaling::Model::DescribeAutoScalingGroupsRequest request;
    Aws::Vector<Aws::String> groupNames;
    groupNames.push_back(groupName);
    request.SetAutoScalingGroupNames(groupNames);

    Aws::AutoScaling::Model::DescribeAutoScalingGroupsOutcome outcome =
            client.DescribeAutoScalingGroups(request);

    if (outcome.IsSuccess()) {
        autoScalingGroup = outcome.GetResult().GetAutoScalingGroups();
    }
    else {
        std::cerr << "Error with AutoScaling::DescribeAutoScalingGroups. "
                  << outcome.GetError().GetMessage()
                  << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Para obtener detalles sobre la API, consulte los siguientes temas en la Referencia de la API de AWS SDK para C++ .
  - CreateAutoScalingGroup

- DeleteAutoScalingGroup
- DescribeAutoScalingGroups
- DescribeAutoScalingInstances
- DescribeScalingActivities
- DisableMetricsCollection
- EnableMetricsCollection
- SetDesiredCapacity
- 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.

```
/**
 * Before running this SDK for Java (v2) code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 *
 * In addition, create a launch template. For more information, see the
 * following topic:
 *
 * https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-launch-
templates.html#create-launch-template
 *
 * This code example performs the following operations:
 * 1. Creates an Auto Scaling group using an AutoScalingWaiter.
 * 2. Gets a specific Auto Scaling group and returns an instance Id value.
```

```
 * 3. Describes Auto Scaling with the Id value.
 * 4. Enables metrics collection.
 * 5. Update an Auto Scaling group.
 * 6. Describes Account details.
 * 7. Describe account details"
 * 8. Updates an Auto Scaling group to use an additional instance.
 * 9. Gets the specific Auto Scaling group and gets the number of instances.
 * 10. List the scaling activities that have occurred for the group.
 * 11. Terminates an instance in the Auto Scaling group.
 * 12. Stops the metrics collection.
 * 13. Deletes the Auto Scaling group.
 */

public class AutoScalingScenario {
    public static final String DASHES = new String(new char[80]).replace("\0",
 "-");
    private static final String ROLES_STACK = "MyCdkAutoScaleStack";
    public static void main(String[] args) throws InterruptedException {
        final String usage = """

                Usage:
                    <groupName>

                Where:
                    groupName - The name of the Auto Scaling group.
                """;

        String groupName = "MyAutoScalingGroup2";
        AutoScalingClient autoScalingClient = AutoScalingClient.builder()
                .region(Region.US_WEST_2)
                .build();

        Ec2Client ec2 = Ec2Client.builder()
                .region(Region.US_WEST_2)
                .build();

        System.out.println(DASHES);
        System.out.println("Welcome to the Amazon EC2 Auto Scaling example
 scenario.");
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("First, we will create a launch template using a
 CloudFormation script");
```

```
        CloudFormationHelper.deployCloudFormationStack(ROLES_STACK);
        Map<String, String> stackOutputs =
CloudFormationHelper.getStackOutputsAsync(ROLES_STACK).join();
        String launchTemplateName = stackOutputs.get("LaunchTemplateNameOutput");
        String vpcZoneId = getVPC(ec2);
        updateTemlate(ec2, launchTemplateName );
        System.out.println("The VPC zone id created by the CloudFormation stack
is"+vpcZoneId);

        System.out.println("1. Create an Auto Scaling group named " + groupName);
        createAutoScalingGroup(autoScalingClient, ec2, groupName,
launchTemplateName, vpcZoneId);

        System.out.println(
                "Wait 1 min for the resources, including the instance. Otherwise,
an empty instance Id is returned");
        Thread.sleep(60000);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("2. Get Auto Scale group Id value");
        String instanceId = getSpecificAutoScalingGroups(autoScalingClient,
groupName);
        if (instanceId.compareTo("") == 0) {
            System.out.println("Error - no instance Id value");
            System.exit(1);
        } else {
            System.out.println("The instance Id value is " + instanceId);
        }
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("3. Describe Auto Scaling with the Id value " +
instanceId);
        describeAutoScalingInstance(autoScalingClient, instanceId);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("4. Enable metrics collection " + instanceId);
        enableMetricsCollection(autoScalingClient, groupName);
        System.out.println(DASHES);

        System.out.println(DASHES);
```

```
        System.out.println("5. Update an Auto Scaling group to update max size to
3");
        updateAutoScalingGroup(autoScalingClient, groupName, launchTemplateName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("6. Describe Auto Scaling groups");
        describeAutoScalingGroups(autoScalingClient, groupName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("7. Describe account details");
        describeAccountLimits(autoScalingClient);
        System.out.println(
                "Wait 1 min for the resources, including the instance. Otherwise,
an empty instance Id is returned");
        Thread.sleep(60000);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("8. Set desired capacity to 2");
        setDesiredCapacity(autoScalingClient, groupName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("9. Get the two instance Id values and state");
        getSpecificAutoScalingGroups(autoScalingClient, groupName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("10. List the scaling activities that have occurred
for the group");
        describeScalingActivities(autoScalingClient, groupName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("11. Terminate an instance in the Auto Scaling
group");
        terminateInstanceInAutoScalingGroup(autoScalingClient, instanceId);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("12. Stop the metrics collection");
        disableMetricsCollection(autoScalingClient, groupName);
```

```
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("13. Delete the Auto Scaling group and cloud formation
resources");
        CloudFormationHelper.destroyCloudFormationStack(ROLES_STACK);
        deleteAutoScalingGroup(autoScalingClient, groupName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("The Scenario has successfully completed.");
        System.out.println(DASHES);

        autoScalingClient.close();
    }

    public static String getVPC(Ec2Client ec2) {
        try {
            DescribeVpcsRequest request = DescribeVpcsRequest.builder()
                    .filters(f -> f.name("isDefault").values("true"))
                    .build();

            DescribeVpcsResponse response = ec2.describeVpcs(request);

            if (!response.vpcs().isEmpty()) {
                Vpc defaultVpc = response.vpcs().get(0);
                System.out.println("Default VPC ID: " + defaultVpc.vpcId());
                return defaultVpc.vpcId();
            } else {
                System.out.println("No default VPC found.");
                return null; // Return null if no default VPC is found
            }

        } catch (Ec2Exception e) {
            System.err.println("EC2 error: " +
e.awsErrorDetails().errorMessage());
            return null; // Return null in case of an error
        }
    }


    public static void updateTemlate(Ec2Client ec2, String launchTemplateName ) {
        // Step 1: Create new launch template version
        String newAmiId = "ami-0025f0db847eb6254";
```

```
        RequestLaunchTemplateData launchTemplateData =
RequestLaunchTemplateData.builder()
                .imageId(newAmiId)
                .build();

        CreateLaunchTemplateVersionRequest createVersionRequest =
CreateLaunchTemplateVersionRequest.builder()
                .launchTemplateName(launchTemplateName)
                .versionDescription("Updated with valid AMI")
                .sourceVersion("1")
                .launchTemplateData(launchTemplateData)
                .build();

        CreateLaunchTemplateVersionResponse createResponse =
ec2.createLaunchTemplateVersion(createVersionRequest);
        int newVersionNumber =
createResponse.launchTemplateVersion().versionNumber().intValue();

        // Step 2: Modify default version
        ModifyLaunchTemplateRequest modifyRequest =
ModifyLaunchTemplateRequest.builder()
                .launchTemplateName(launchTemplateName)
                .defaultVersion(String.valueOf(newVersionNumber))
                .build();

        ec2.modifyLaunchTemplate(modifyRequest);
        System.out.println("Updated launch template to version " +
newVersionNumber + " with AMI " + newAmiId);
    }


    public static void describeScalingActivities(AutoScalingClient
autoScalingClient, String groupName) {
        try {
            DescribeScalingActivitiesRequest scalingActivitiesRequest =
DescribeScalingActivitiesRequest.builder()
                    .autoScalingGroupName(groupName)
                    .maxRecords(10)
                    .build();

            DescribeScalingActivitiesResponse response = autoScalingClient
                    .describeScalingActivities(scalingActivitiesRequest);
            List<Activity> activities = response.activities();
            for (Activity activity : activities) {
```

```
                System.out.println("The activity Id is " +
 activity.activityId());
                System.out.println("The activity details are " +
 activity.details());
            }

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

    public static void setDesiredCapacity(AutoScalingClient autoScalingClient,
 String groupName) {
        try {
            SetDesiredCapacityRequest capacityRequest =
 SetDesiredCapacityRequest.builder()
                    .autoScalingGroupName(groupName)
                    .desiredCapacity(2)
                    .build();

            autoScalingClient.setDesiredCapacity(capacityRequest);
            System.out.println("You have set the DesiredCapacity to 2");

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

    public static void createAutoScalingGroup(AutoScalingClient
 autoScalingClient,
                                               Ec2Client ec2Client,
                                               String groupName,
                                               String launchTemplateName,
                                               String vpcId) {
        try {
            // Step 1: Get one subnet ID in the given VPC
            DescribeSubnetsRequest subnetRequest =
 DescribeSubnetsRequest.builder()
                    .filters(Filter.builder().name("vpc-
id").values(vpcId).build())
                    .build();
```

```
            DescribeSubnetsResponse subnetResponse =
ec2Client.describeSubnets(subnetRequest);

            if (subnetResponse.subnets().isEmpty()) {
                throw new RuntimeException("No subnets found in VPC: " + vpcId);
            }

            String subnetId = subnetResponse.subnets().get(0).subnetId(); // Use
first subnet
            System.out.println("Using subnet: " + subnetId);

            // Step 2: Create launch template reference
            LaunchTemplateSpecification templateSpecification =
LaunchTemplateSpecification.builder()
                    .launchTemplateName(launchTemplateName)
                    .build();

            // Step 3: Create Auto Scaling group
            CreateAutoScalingGroupRequest request =
CreateAutoScalingGroupRequest.builder()
                    .autoScalingGroupName(groupName)
                    .launchTemplate(templateSpecification)
                    .minSize(1)
                    .maxSize(1)
                    .vpcZoneIdentifier(subnetId)  // Correct: subnet ID, not VPC
ID
                    .build();

            autoScalingClient.createAutoScalingGroup(request);

            // Step 4: Wait until group is created
            AutoScalingWaiter waiter = autoScalingClient.waiter();
            DescribeAutoScalingGroupsRequest groupsRequest =
DescribeAutoScalingGroupsRequest.builder()
                    .autoScalingGroupNames(groupName)
                    .build();

            WaiterResponse<DescribeAutoScalingGroupsResponse> waiterResponse =
                    waiter.waitUntilGroupExists(groupsRequest);

            waiterResponse.matched().response().ifPresent(System.out::println);
            System.out.println("Auto Scaling Group created");

        } catch (Ec2Exception | AutoScalingException e) {
```

```
            System.err.println("Error: " + e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static void describeAutoScalingInstance(AutoScalingClient
autoScalingClient, String id) {
        try {
            DescribeAutoScalingInstancesRequest
describeAutoScalingInstancesRequest = DescribeAutoScalingInstancesRequest
                    .builder()
                    .instanceIds(id)
                    .build();

            DescribeAutoScalingInstancesResponse response = autoScalingClient

.describeAutoScalingInstances(describeAutoScalingInstancesRequest);
            List<AutoScalingInstanceDetails> instances =
response.autoScalingInstances();
            for (AutoScalingInstanceDetails instance : instances) {
                System.out.println("The instance lifecycle state is: " +
instance.lifecycleState());
            }

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

    public static void describeAutoScalingGroups(AutoScalingClient
autoScalingClient, String groupName) {
        try {
            DescribeAutoScalingGroupsRequest groupsRequest =
DescribeAutoScalingGroupsRequest.builder()
                    .autoScalingGroupNames(groupName)
                    .maxRecords(10)
                    .build();

            DescribeAutoScalingGroupsResponse response =
autoScalingClient.describeAutoScalingGroups(groupsRequest);
            List<AutoScalingGroup> groups = response.autoScalingGroups();
            for (AutoScalingGroup group : groups) {
```

```
                System.out.println("*** The service to use for the health checks:
" + group.healthCheckType());
            }

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

    public static String getSpecificAutoScalingGroups(AutoScalingClient
autoScalingClient, String groupName) {
        try {
            String instanceId = "";
            DescribeAutoScalingGroupsRequest scalingGroupsRequest =
DescribeAutoScalingGroupsRequest.builder()
                    .autoScalingGroupNames(groupName)
                    .build();

            DescribeAutoScalingGroupsResponse response = autoScalingClient
                    .describeAutoScalingGroups(scalingGroupsRequest);
            List<AutoScalingGroup> groups = response.autoScalingGroups();
            for (AutoScalingGroup group : groups) {
                System.out.println("The group name is " +
group.autoScalingGroupName());
                System.out.println("The group ARN is " +
group.autoScalingGroupARN());
                List<Instance> instances = group.instances();

                for (Instance instance : instances) {
                    instanceId = instance.instanceId();
                    System.out.println("The instance id is " + instanceId);
                    System.out.println("The lifecycle state is " +
instance.lifecycleState());
                }
            }

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

```
   public static void enableMetricsCollection(AutoScalingClient
autoScalingClient, String groupName) {
       try {
           EnableMetricsCollectionRequest collectionRequest =
EnableMetricsCollectionRequest.builder()
                   .autoScalingGroupName(groupName)
                   .metrics("GroupMaxSize")
                   .granularity("1Minute")
                   .build();

           autoScalingClient.enableMetricsCollection(collectionRequest);
           System.out.println("The enable metrics collection operation was
successful");

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

   public static void disableMetricsCollection(AutoScalingClient
autoScalingClient, String groupName) {
       try {
           DisableMetricsCollectionRequest disableMetricsCollectionRequest =
DisableMetricsCollectionRequest.builder()
                   .autoScalingGroupName(groupName)
                   .metrics("GroupMaxSize")
                   .build();


autoScalingClient.disableMetricsCollection(disableMetricsCollectionRequest);
           System.out.println("The disable metrics collection operation was
successful");

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

   public static void describeAccountLimits(AutoScalingClient autoScalingClient)
{
       try {
```

```
            DescribeAccountLimitsResponse response =
autoScalingClient.describeAccountLimits();
            System.out.println("The max number of auto scaling groups is " +
response.maxNumberOfAutoScalingGroups());
            System.out.println("The current number of auto scaling groups is " +
response.numberOfAutoScalingGroups());

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

    public static void updateAutoScalingGroup(AutoScalingClient
autoScalingClient, String groupName,
            String launchTemplateName) {
        try {
            AutoScalingWaiter waiter = autoScalingClient.waiter();
            LaunchTemplateSpecification templateSpecification =
LaunchTemplateSpecification.builder()
                    .launchTemplateName(launchTemplateName)
                    .build();

            UpdateAutoScalingGroupRequest groupRequest =
UpdateAutoScalingGroupRequest.builder()
                    .maxSize(3)
                    .autoScalingGroupName(groupName)
                    .launchTemplate(templateSpecification)
                    .build();

            autoScalingClient.updateAutoScalingGroup(groupRequest);
            DescribeAutoScalingGroupsRequest groupsRequest =
DescribeAutoScalingGroupsRequest.builder()
                    .autoScalingGroupNames(groupName)
                    .build();

            WaiterResponse<DescribeAutoScalingGroupsResponse> waiterResponse =
waiter
                    .waitUntilGroupInService(groupsRequest);
            waiterResponse.matched().response().ifPresent(System.out::println);
            System.out.println("You successfully updated the auto scaling group
" + groupName);

        } catch (AutoScalingException e) {
```

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

    public static void terminateInstanceInAutoScalingGroup(AutoScalingClient
 autoScalingClient, String instanceId) {
        try {
            TerminateInstanceInAutoScalingGroupRequest request =
 TerminateInstanceInAutoScalingGroupRequest.builder()
                    .instanceId(instanceId)
                    .shouldDecrementDesiredCapacity(false)
                    .build();

            autoScalingClient.terminateInstanceInAutoScalingGroup(request);
            System.out.println("You have terminated instance " + instanceId);

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

    public static void deleteAutoScalingGroup(AutoScalingClient
 autoScalingClient, String groupName) {
        try {
            DeleteAutoScalingGroupRequest deleteAutoScalingGroupRequest =
 DeleteAutoScalingGroupRequest.builder()
                    .autoScalingGroupName(groupName)
                    .forceDelete(true)
                    .build();


 autoScalingClient.deleteAutoScalingGroup(deleteAutoScalingGroupRequest);
            System.out.println("You successfully deleted " + groupName);

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

- Para obtener detalles sobre la API, consulte los siguientes temas en la Referencia de la API de AWS SDK for Java 2.x .
  - CreateAutoScalingGroup
  - DeleteAutoScalingGroup
  - DescribeAutoScalingGroups
  - DescribeAutoScalingInstances
  - DescribeScalingActivities
  - DisableMetricsCollection
  - EnableMetricsCollection
  - SetDesiredCapacity
  - TerminateInstanceInAutoScalingGroup
  - UpdateAutoScalingGroup

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.

```
suspend fun main(args: Array<String>) {
    val usage = """
    Usage:
        <groupName> <launchTemplateName> <serviceLinkedRoleARN> <vpcZoneId>

    Where:
        groupName - The name of the Auto Scaling group.
        launchTemplateName - The name of the launch template.
        serviceLinkedRoleARN - The Amazon Resource Name (ARN) of the service-
linked role that the Auto Scaling group uses.
        vpcZoneId - A subnet Id for a virtual private cloud (VPC) where instances
 in the Auto Scaling group can be created.
    """
```

```
    if (args.size != 4) {
        println(usage)
        exitProcess(1)
    }

    val groupName = args[0]
    val launchTemplateName = args[1]
    val serviceLinkedRoleARN = args[2]
    val vpcZoneId = args[3]

    println("**** Create an Auto Scaling group named $groupName")
    createAutoScalingGroup(groupName, launchTemplateName, serviceLinkedRoleARN,
vpcZoneId)

    println("Wait 1 min for the resources, including the instance. Otherwise, an
empty instance Id is returned")
    delay(60000)

    val instanceId = getSpecificAutoScaling(groupName)
    if (instanceId.compareTo("") == 0) {
        println("Error - no instance Id value")
        exitProcess(1)
    } else {
        println("The instance Id value is $instanceId")
    }

    println("**** Describe Auto Scaling with the Id value $instanceId")
    describeAutoScalingInstance(instanceId)

    println("**** Enable metrics collection $instanceId")
    enableMetricsCollection(groupName)

    println("**** Update an Auto Scaling group to maximum size of 3")
    updateAutoScalingGroup(groupName, launchTemplateName, serviceLinkedRoleARN)

    println("**** Describe all Auto Scaling groups to show the current state of
the groups")
    describeAutoScalingGroups(groupName)

    println("**** Describe account details")
    describeAccountLimits()

    println("Wait 1 min for the resources, including the instance. Otherwise, an
empty instance Id is returned")
```

```
    delay(60000)

    println("**** Set desired capacity to 2")
    setDesiredCapacity(groupName)

    println("**** Get the two instance Id values and state")
    getAutoScalingGroups(groupName)

    println("**** List the scaling activities that have occurred for the group")
    describeScalingActivities(groupName)

    println("**** Terminate an instance in the Auto Scaling group")
    terminateInstanceInAutoScalingGroup(instanceId)

    println("**** Stop the metrics collection")
    disableMetricsCollection(groupName)

    println("**** Delete the Auto Scaling group")
    deleteSpecificAutoScalingGroup(groupName)
}

suspend fun describeAutoScalingGroups(groupName: String) {
    val groupsReques =
        DescribeAutoScalingGroupsRequest {
            autoScalingGroupNames = listOf(groupName)
            maxRecords = 10
        }

    AutoScalingClient { region = "us-east-1" }.use { autoScalingClient ->
        val response = autoScalingClient.describeAutoScalingGroups(groupsReques)
        response.autoScalingGroups?.forEach { group ->
            println("The service to use for the health checks:
 ${group.healthCheckType}")
        }
    }
}

suspend fun disableMetricsCollection(groupName: String) {
    val disableMetricsCollectionRequest =
        DisableMetricsCollectionRequest {
            autoScalingGroupName = groupName
            metrics = listOf("GroupMaxSize")
        }
```

```
    AutoScalingClient { region = "us-east-1" }.use { autoScalingClient ->

 autoScalingClient.disableMetricsCollection(disableMetricsCollectionRequest)
        println("The disable metrics collection operation was successful")
    }
}

suspend fun describeScalingActivities(groupName: String?) {
    val scalingActivitiesRequest =
        DescribeScalingActivitiesRequest {
            autoScalingGroupName = groupName
            maxRecords = 10
        }

    AutoScalingClient { region = "us-east-1" }.use { autoScalingClient ->
        val response =
 autoScalingClient.describeScalingActivities(scalingActivitiesRequest)
        response.activities?.forEach { activity ->
            println("The activity Id is ${activity.activityId}")
            println("The activity details are ${activity.details}")
        }
    }
}

suspend fun getAutoScalingGroups(groupName: String) {
    val scalingGroupsRequest =
        DescribeAutoScalingGroupsRequest {
            autoScalingGroupNames = listOf(groupName)
        }

    AutoScalingClient { region = "us-east-1" }.use { autoScalingClient ->
        val response =
 autoScalingClient.describeAutoScalingGroups(scalingGroupsRequest)
        response.autoScalingGroups?.forEach { group ->
            println("The group name is ${group.autoScalingGroupName}")
            println("The group ARN is ${group.autoScalingGroupArn}")
            group.instances?.forEach { instance ->
                println("The instance id is ${instance.instanceId}")
                println("The lifecycle state is " + instance.lifecycleState)
            }
        }
    }
}
```

```
suspend fun setDesiredCapacity(groupName: String) {
    val capacityRequest =
        SetDesiredCapacityRequest {
            autoScalingGroupName = groupName
            desiredCapacity = 2
        }

    AutoScalingClient { region = "us-east-1" }.use { autoScalingClient ->
        autoScalingClient.setDesiredCapacity(capacityRequest)
        println("You set the DesiredCapacity to 2")
    }
}

suspend fun updateAutoScalingGroup(
    groupName: String,
    launchTemplateNameVal: String,
    serviceLinkedRoleARNVal: String,
) {
    val templateSpecification =
        LaunchTemplateSpecification {
            launchTemplateName = launchTemplateNameVal
        }

    val groupRequest =
        UpdateAutoScalingGroupRequest {
            maxSize = 3
            serviceLinkedRoleArn = serviceLinkedRoleARNVal
            autoScalingGroupName = groupName
            launchTemplate = templateSpecification
        }

    val groupsRequestWaiter =
        DescribeAutoScalingGroupsRequest {
            autoScalingGroupNames = listOf(groupName)
        }

    AutoScalingClient { region = "us-east-1" }.use { autoScalingClient ->
        autoScalingClient.updateAutoScalingGroup(groupRequest)
        autoScalingClient.waitUntilGroupExists(groupsRequestWaiter)
        println("You successfully updated the Auto Scaling group  $groupName")
    }
}

suspend fun createAutoScalingGroup(
```

```
    groupName: String,
    launchTemplateNameVal: String,
    serviceLinkedRoleARNVal: String,
    vpcZoneIdVal: String,
) {
    val templateSpecification =
        LaunchTemplateSpecification {
            launchTemplateName = launchTemplateNameVal
        }

    val request =
        CreateAutoScalingGroupRequest {
            autoScalingGroupName = groupName
            availabilityZones = listOf("us-east-1a")
            launchTemplate = templateSpecification
            maxSize = 1
            minSize = 1
            vpcZoneIdentifier = vpcZoneIdVal
            serviceLinkedRoleArn = serviceLinkedRoleARNVal
        }

    // This object is required for the waiter call.
    val groupsRequestWaiter =
        DescribeAutoScalingGroupsRequest {
            autoScalingGroupNames = listOf(groupName)
        }

    AutoScalingClient { region = "us-east-1" }.use { autoScalingClient ->
        autoScalingClient.createAutoScalingGroup(request)
        autoScalingClient.waitUntilGroupExists(groupsRequestWaiter)
        println("$groupName was created!")
    }
}

suspend fun describeAutoScalingInstance(id: String) {
    val describeAutoScalingInstancesRequest =
        DescribeAutoScalingInstancesRequest {
            instanceIds = listOf(id)
        }

    AutoScalingClient { region = "us-east-1" }.use { autoScalingClient ->
        val response =
 autoScalingClient.describeAutoScalingInstances(describeAutoScalingInstancesRequest)
        response.autoScalingInstances?.forEach { group ->
```

```
            println("The instance lifecycle state is: ${group.lifecycleState}")
        }
    }
}

suspend fun enableMetricsCollection(groupName: String?) {
    val collectionRequest =
        EnableMetricsCollectionRequest {
            autoScalingGroupName = groupName
            metrics = listOf("GroupMaxSize")
            granularity = "1Minute"
        }

    AutoScalingClient { region = "us-east-1" }.use { autoScalingClient ->
        autoScalingClient.enableMetricsCollection(collectionRequest)
        println("The enable metrics collection operation was successful")
    }
}

suspend fun getSpecificAutoScaling(groupName: String): String {
    var instanceId = ""
    val scalingGroupsRequest =
        DescribeAutoScalingGroupsRequest {
            autoScalingGroupNames = listOf(groupName)
        }

    AutoScalingClient { region = "us-east-1" }.use { autoScalingClient ->
        val response =
 autoScalingClient.describeAutoScalingGroups(scalingGroupsRequest)
        response.autoScalingGroups?.forEach { group ->
            println("The group name is ${group.autoScalingGroupName}")
            println("The group ARN is ${group.autoScalingGroupArn}")

            group.instances?.forEach { instance ->
                instanceId = instance.instanceId.toString()
            }
        }
    }
    return instanceId
}

suspend fun describeAccountLimits() {
    AutoScalingClient { region = "us-east-1" }.use { autoScalingClient ->
```

```
        val response =
 autoScalingClient.describeAccountLimits(DescribeAccountLimitsRequest {})
        println("The max number of Auto Scaling groups is
 ${response.maxNumberOfAutoScalingGroups}")
        println("The current number of Auto Scaling groups is
 ${response.numberOfAutoScalingGroups}")
    }
}

suspend fun terminateInstanceInAutoScalingGroup(instanceIdVal: String) {
    val request =
        TerminateInstanceInAutoScalingGroupRequest {
            instanceId = instanceIdVal
            shouldDecrementDesiredCapacity = false
        }

    AutoScalingClient { region = "us-east-1" }.use { autoScalingClient ->
        autoScalingClient.terminateInstanceInAutoScalingGroup(request)
        println("You have terminated instance $instanceIdVal")
    }
}

suspend fun deleteSpecificAutoScalingGroup(groupName: String) {
    val deleteAutoScalingGroupRequest =
        DeleteAutoScalingGroupRequest {
            autoScalingGroupName = groupName
            forceDelete = true
        }

    AutoScalingClient { region = "us-east-1" }.use { autoScalingClient ->
        autoScalingClient.deleteAutoScalingGroup(deleteAutoScalingGroupRequest)
        println("You successfully deleted $groupName")
    }
}
```

- Para obtener detalles sobre la API, consulte los siguientes temas en la Referencia de la API de AWS SDK para Kotlin.
  - CreateAutoScalingGroup
  - DeleteAutoScalingGroup
  - DescribeAutoScalingGroups
  - DescribeAutoScalingInstances

- DescribeScalingActivities
- DisableMetricsCollection
- EnableMetricsCollection
- SetDesiredCapacity
- TerminateInstanceInAutoScalingGroup
- UpdateAutoScalingGroup

PHP

SDK para PHP

> **Note**
>
> Hay más información GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
namespace AutoScaling;

use Aws\AutoScaling\AutoScalingClient;
use Aws\CloudWatch\CloudWatchClient;
use Aws\Ec2\Ec2Client;
use AwsUtilities\AWSServiceClass;
use AwsUtilities\RunnableExample;

class GettingStartedWithAutoScaling implements RunnableExample
{
    protected Ec2Client $ec2Client;
    protected AutoScalingClient $autoScalingClient;
    protected AutoScalingService $autoScalingService;
    protected CloudWatchClient $cloudWatchClient;
    protected string $templateName;
    protected string $autoScalingGroupName;
    protected array $role;

    public function runExample()
    {
        echo("\n");
        echo("--------------------------------------\n");
```

```
        print("Welcome to the Amazon EC2 Auto Scaling getting started demo using
 PHP!\n");
        echo("-------------------------------------\n");

        $clientArgs = [
            'region' => 'us-west-2',
            'version' => 'latest',
            'profile' => 'default',
        ];
        $uniqid = uniqid();

        $this->autoScalingClient = new AutoScalingClient($clientArgs);
        $this->autoScalingService = new AutoScalingService($this-
>autoScalingClient);
        $this->cloudWatchClient = new CloudWatchClient($clientArgs);

        AWSServiceClass::$waitTime = 5;
        AWSServiceClass::$maxWaitAttempts = 20;

        /**
         * Step 0: Create an EC2 launch template that you'll use to create an
 Auto Scaling group.
         */
        $this->ec2Client = new EC2Client($clientArgs);
        $this->templateName = "example_launch_template_$uniqid";
        $instanceType = "t1.micro";
        $amiId = "ami-0ca285d4c2cda3300";
        $launchTemplate = $this->ec2Client->createLaunchTemplate(
            [
            'LaunchTemplateName' => $this->templateName,
            'LaunchTemplateData' => [
                'InstanceType' => $instanceType,
                'ImageId' => $amiId,
            ]
            ]
        );

        /**
         * Step 1: CreateAutoScalingGroup: pass it the launch template you
 created in step 0.
         */
        $availabilityZones[] = $this->ec2Client->describeAvailabilityZones([])
['AvailabilityZones'][1]['ZoneName'];
```

```
        $this->autoScalingGroupName = "demoAutoScalingGroupName_$uniqid";
        $minSize = 1;
        $maxSize = 1;
        $launchTemplateId = $launchTemplate['LaunchTemplate']
['LaunchTemplateId'];
        $this->autoScalingService->createAutoScalingGroup(
            $this->autoScalingGroupName,
            $availabilityZones,
            $minSize,
            $maxSize,
            $launchTemplateId
        );

        $this->autoScalingService->waitUntilGroupInService([$this-
>autoScalingGroupName]);
        $autoScalingGroup = $this->autoScalingService-
>describeAutoScalingGroups([$this->autoScalingGroupName]);

        /**
         * Step 2: DescribeAutoScalingInstances: show that one instance has
 launched.
         */
        $instanceIds = [$autoScalingGroup['AutoScalingGroups'][0]['Instances'][0]
['InstanceId']];
        $instances = $this->autoScalingService-
>describeAutoScalingInstances($instanceIds);
        echo "The Auto Scaling group {$this->autoScalingGroupName} was created
 successfully.\n";
        echo count($instances['AutoScalingInstances']) . " instances were created
 for the group.\n";
        echo $autoScalingGroup['AutoScalingGroups'][0]['MaxSize'] . " is the max
 number of instances for the group.\n";

        /**
         * Step 3: EnableMetricsCollection: enable all metrics or a subset.
         */
        $this->autoScalingService->enableMetricsCollection($this-
>autoScalingGroupName, "1Minute");

        /**
         * Step 4: UpdateAutoScalingGroup: update max size to 3.
         */
        echo "Updating the max number of instances to 3.\n";
```

```
        $this->autoScalingService->updateAutoScalingGroup($this-
>autoScalingGroupName, ['MaxSize' => 3]);

        /**
         * Step 5: DescribeAutoScalingGroups: show the current state of the
 group.
         */
        $autoScalingGroup = $this->autoScalingService-
>describeAutoScalingGroups([$this->autoScalingGroupName]);
        echo $autoScalingGroup['AutoScalingGroups'][0]['MaxSize'];
        echo " is the updated max number of instances for the group.\n";

        $limits = $this->autoScalingService->describeAccountLimits();
        echo "Here are your account limits:\n";
        echo "MaxNumberOfAutoScalingGroups:
 {$limits['MaxNumberOfAutoScalingGroups']}\n";
        echo "MaxNumberOfLaunchConfigurations:
 {$limits['MaxNumberOfLaunchConfigurations']}\n";
        echo "NumberOfAutoScalingGroups:
 {$limits['NumberOfAutoScalingGroups']}\n";
        echo "NumberOfLaunchConfigurations:
 {$limits['NumberOfLaunchConfigurations']}\n";

        /**
         * Step 6: SetDesiredCapacity: set desired capacity to 2.
         */
        $this->autoScalingService->setDesiredCapacity($this-
>autoScalingGroupName, 2);
        sleep(10); // Wait for the group to start processing the request.
        $this->autoScalingService->waitUntilGroupInService([$this-
>autoScalingGroupName]);

        /**
         * Step 7: DescribeAutoScalingInstances: show that two instances are
 launched.
         */
        $autoScalingGroups = $this->autoScalingService-
>describeAutoScalingGroups([$this->autoScalingGroupName]);
        foreach ($autoScalingGroups['AutoScalingGroups'] as $autoScalingGroup) {
            echo "There is a group named:
 {$autoScalingGroup['AutoScalingGroupName']}";
            echo "with an ARN of {$autoScalingGroup['AutoScalingGroupARN']}.\n";
            foreach ($autoScalingGroup['Instances'] as $instance) {
```

```
                echo "{$autoScalingGroup['AutoScalingGroupName']} has an instance
 with id of: ";
                echo "{$instance['InstanceId']} and a lifecycle state of:
 {$instance['LifecycleState']}.\n";
            }
        }

        /**
         * Step 8: TerminateInstanceInAutoScalingGroup: terminate one of the
 instances in the group.
         */
        $this->autoScalingService-
>terminateInstanceInAutoScalingGroup($instance['InstanceId'], false);
        do {
            sleep(10);
            $instances = $this->autoScalingService-
>describeAutoScalingInstances([$instance['InstanceId']]);
        } while (count($instances['AutoScalingInstances']) > 0);
        do {
            sleep(10);
            $autoScalingGroups = $this->autoScalingService-
>describeAutoScalingGroups([$this->autoScalingGroupName]);
            $instances = $autoScalingGroups['AutoScalingGroups'][0]['Instances'];
        } while (count($instances) < 2);
        $this->autoScalingService->waitUntilGroupInService([$this-
>autoScalingGroupName]);
        foreach ($autoScalingGroups['AutoScalingGroups'] as $autoScalingGroup) {
            echo "There is a group named:
 {$autoScalingGroup['AutoScalingGroupName']}";
            echo "with an ARN of {$autoScalingGroup['AutoScalingGroupARN']}.\n";
            foreach ($autoScalingGroup['Instances'] as $instance) {
                echo "{$autoScalingGroup['AutoScalingGroupName']} has an instance
 with id of: ";
                echo "{$instance['InstanceId']} and a lifecycle state of:
 {$instance['LifecycleState']}.\n";
            }
        }

        /**
         * Step 9: DescribeScalingActivities: list the scaling activities that
 have occurred for the group so far.
         */
        $activities = $this->autoScalingService-
>describeScalingActivities($autoScalingGroup['AutoScalingGroupName']);
```

```
        echo "We found " . count($activities['Activities']) . " activities.\n";
        foreach ($activities['Activities'] as $activity) {
            echo "{$activity['ActivityId']} - {$activity['StartTime']} -
{$activity['Description']}\n";
        }

        /**
         * Step 10: Use the Amazon CloudWatch API to get and show some metrics
collected for the group.
         */
        $metricsNamespace = 'AWS/AutoScaling';
        $metricsDimensions = [
            [
                'Name' => 'AutoScalingGroupName',
                'Value' => $autoScalingGroup['AutoScalingGroupName'],
            ],
        ];
        $metrics = $this->cloudWatchClient->listMetrics(
            [
            'Dimensions' => $metricsDimensions,
            'Namespace' => $metricsNamespace,
            ]
        );
        foreach ($metrics['Metrics'] as $metric) {
            $timespan = 5;
            if ($metric['MetricName'] != 'GroupTotalCapacity' &&
$metric['MetricName'] != 'GroupMaxSize') {
                continue;
            }
            echo "Over the last $timespan minutes, {$metric['MetricName']}
recorded:\n";
            $stats = $this->cloudWatchClient->getMetricStatistics(
                [
                'Dimensions' => $metricsDimensions,
                'EndTime' => time(),
                'StartTime' => time() - (5 * 60),
                'MetricName' => $metric['MetricName'],
                'Namespace' => $metricsNamespace,
                'Period' => 60,
                'Statistics' => ['Sum'],
                ]
            );
            foreach ($stats['Datapoints'] as $stat) {
                echo "{$stat['Timestamp']}: {$stat['Sum']}\n";
```

```
            }
        }

        return $instances;
    }

    public function cleanUp()
    {
        /**
         * Step 11: DisableMetricsCollection: disable all metrics.
         */
        $this->autoScalingService->disableMetricsCollection($this-
>autoScalingGroupName);

        /**
         * Step 12: DeleteAutoScalingGroup: to delete the group you must stop all
 instances.
         * - UpdateAutoScalingGroup with MinSize=0
         * - TerminateInstanceInAutoScalingGroup for each instance,
         *     specify ShouldDecrementDesiredCapacity=True. Wait for instances to
 stop.
         * - Now you can delete the group.
         */
        $this->autoScalingService->updateAutoScalingGroup($this-
>autoScalingGroupName, ['MinSize' => 0]);
        $this->autoScalingService->terminateAllInstancesInAutoScalingGroup($this-
>autoScalingGroupName);
        $this->autoScalingService->waitUntilGroupInService([$this-
>autoScalingGroupName]);
        $this->autoScalingService->deleteAutoScalingGroup($this-
>autoScalingGroupName);

        /**
         * Step 13: Delete launch template.
         */
        $this->ec2Client->deleteLaunchTemplate(
            [
            'LaunchTemplateName' => $this->templateName,
            ]
        );
    }

    public function helloService()
    {
```

```
        $autoScalingClient = new AutoScalingClient([
            'region' => 'us-west-2',
            'version' => 'latest',
            'profile' => 'default',
        ]);

        $groups = $autoScalingClient->describeAutoScalingGroups([]);
        var_dump($groups);
    }
}
```

- Para obtener detalles sobre la API, consulte los siguientes temas en la Referencia de la API de AWS SDK para PHP .
  - CreateAutoScalingGroup
  - DeleteAutoScalingGroup
  - DescribeAutoScalingGroups
  - DescribeAutoScalingInstances
  - DescribeScalingActivities
  - DisableMetricsCollection
  - EnableMetricsCollection
  - SetDesiredCapacity
  - 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.

Ejecutar un escenario interactivo en un símbolo del sistema.

```
def run_scenario(as_wrapper: AutoScalingWrapper, svc_helper: ServiceHelper) ->
 None:
    """
    Runs the scenario demonstrating the management of Auto Scaling groups and
 instances.

    :param as_wrapper: An instance of the AutoScalingWrapper that manages Auto
 Scaling groups.
    :param svc_helper: An instance of the ServiceHelper that interacts with AWS
 services.
    :return: None
    """
    logging.basicConfig(level=logging.INFO, format="%(levelname)s: %(message)s")

    logger.info("Starting the Amazon EC2 Auto Scaling demo.")

    print("-" * 88)
    print(
        "Welcome to the Amazon EC2 Auto Scaling demo for managing groups and
 instances."
    )
    print("-" * 88)

    print(
        "This example requires a launch template that specifies how to create "
        "EC2 instances. You can use an existing template or create a new one."
    )
    template_name = q.ask(
        "Enter the name of an existing launch template or press Enter to create a
 new one: "
    )
    template = None
    if template_name:
        template = svc_helper.get_template(template_name)
    if template is None:
        inst_type = "t1.micro"
        ami_id = "ami-0ca285d4c2cda3300"
        print("Let's create a launch template with the following
 specifications:")
        print(f"\tInstanceType: {inst_type}")
        print(f"\tAMI ID: {ami_id}")
        template_name = q.ask("Enter a name for the template: ", q.non_empty)
        template = svc_helper.create_template(template_name, inst_type, ami_id)
```

```
    print("-" * 88)

    print("Let's create an Auto Scaling group.")
    group_name = q.ask("Enter a name for the group: ", q.non_empty)
    zones = svc_helper.get_availability_zones()
    print("EC2 instances can be created in the following Availability Zones:")
    for index, zone in enumerate(zones):
        print(f"\t{index+1}. {zone}")
    print(f"\t{len(zones)+1}. All zones")
    zone_sel = q.ask(
        "Which zone do you want to use? ", q.is_int, q.in_range(1, len(zones) +
1)
    )
    group_zones = [zones[zone_sel - 1]] if zone_sel <= len(zones) else zones
    print(f"Creating group {group_name}...")
    as_wrapper.create_autoscaling_group(group_name, group_zones, template_name,
1, 1)
    wait(10)
    group = as_wrapper.describe_group(group_name)
    logger.info("Created Auto Scaling group %s.", group_name)
    print("Created group:")
    pp(group)
    print("Waiting for instance to start...")
    wait_for_group(group_name, as_wrapper)
    print("-" * 88)

    use_metrics = q.ask(
        "Do you want to collect metrics about Amazon EC2 Auto Scaling during this
demo (y/n)? ",
        q.is_yesno,
    )
    if use_metrics:
        as_wrapper.enable_metrics(
            group_name,
            [
                "GroupMinSize",
                "GroupMaxSize",
                "GroupDesiredCapacity",
                "GroupInServiceInstances",
                "GroupTotalInstances",
            ],
        )
        logger.info("Enabled metrics for Auto Scaling group %s.", group_name)
        print(f"Metrics enabled for {group_name}.")
```

```
    print("-" * 88)

    print(f"Let's update the maximum number of instances in {group_name} from 1
to 3.")
    q.ask("Press Enter when you're ready.")
    as_wrapper.update_group(group_name, MaxSize=3)
    group = as_wrapper.describe_group(group_name)
    logger.info("Updated maximum size for group %s to 3.", group_name)
    print("The group still has one running instance, but can have up to three:")
    print_simplified_group(group)
    print("-" * 88)

    print(f"Let's update the desired capacity of {group_name} from 1 to 2.")
    q.ask("Press Enter when you're ready.")
    as_wrapper.set_desired_capacity(group_name, 2)
    wait(10)
    group = as_wrapper.describe_group(group_name)
    logger.info("Set desired capacity for group %s to 2.", group_name)
    print("Here's the current state of the group:")
    print_simplified_group(group)
    print("-" * 88)
    print("Waiting for the new instance to start...")
    instance_ids = wait_for_group(group_name, as_wrapper)
    print("-" * 88)

    print(f"Let's terminate one of the instances in {group_name}.")
    print("Because the desired capacity is 2, another instance will start.")
    print("The currently running instances are:")
    for index, inst_id in enumerate(instance_ids):
        print(f"\t{index+1}. {inst_id}")
    inst_sel = q.ask(
        "Which instance do you want to stop? ",
        q.is_int,
        q.in_range(1, len(instance_ids) + 1),
    )
    print(f"Stopping {instance_ids[inst_sel-1]}...")
    as_wrapper.terminate_instance(instance_ids[inst_sel - 1], False)
    wait(10)
    group = as_wrapper.describe_group(group_name)
    logger.info(
        "Terminated instance %s in group %s.", instance_ids[inst_sel - 1],
group_name
    )
    print(f"Here's the state of {group_name}:")
```

```
    print_simplified_group(group)
    print("Waiting for the scaling activities to complete...")
    wait_for_group(group_name, as_wrapper)
    print("-" * 88)

    print(f"Let's get a report of scaling activities for {group_name}.")
    q.ask("Press Enter when you're ready.")
    activities = as_wrapper.describe_scaling_activities(group_name)
    logger.info(
        "Retrieved %d scaling activities for group %s.", len(activities),
group_name
    )
    print(
        f"Found {len(activities)} activities.\n"
        f"Activities are ordered with the most recent one first:"
    )
    for act in activities:
        pp(act)
    print("-" * 88)

    if use_metrics:
        print("Let's look at CloudWatch metrics.")
        metric_namespace = "AWS/AutoScaling"
        metric_dimensions = [{"Name": "AutoScalingGroupName", "Value":
group_name}]
        print(f"The following metrics are enabled for {group_name}:")
        done = False
        while not done:
            metrics = svc_helper.get_metrics(metric_namespace, metric_dimensions)
            for index, metric in enumerate(metrics):
                print(f"\t{index+1}. {metric.name}")
            print(f"\t{len(metrics)+1}. None")
            metric_sel = q.ask(
                "Which metric do you want to see? ",
                q.is_int,
                q.in_range(1, len(metrics) + 1),
            )
            if metric_sel < len(metrics) + 1:
                span = 5
                metric = metrics[metric_sel - 1]
                print(f"Over the last {span} minutes, {metric.name} recorded:")
                # CloudWatch metric times are in the UTC+0 time zone.
                now = datetime.now(timezone.utc)
                metric_data = svc_helper.get_metric_statistics(
```

```
                    metric_dimensions, metric, now - timedelta(minutes=span), now
                )
                pp(metric_data)
                if not q.ask("Do you want to see another metric (y/n)? ",
q.is_yesno):
                    done = True
            else:
                done = True

    print(f"Let's clean up.")
    q.ask("Press Enter when you're ready.")
    if use_metrics:
        print(f"Stopping metrics collection for {group_name}.")
        as_wrapper.disable_metrics(group_name)
        logger.info("Disabled metrics collection for group %s.", group_name)

    print(
        "You must terminate all instances in the group before you can delete the
group."
    )
    print("Set minimum size to 0.")
    as_wrapper.update_group(group_name, MinSize=0)
    group = as_wrapper.describe_group(group_name)
    instance_ids = [inst["InstanceId"] for inst in group["Instances"]]
    for inst_id in instance_ids:
        print(f"Stopping {inst_id}.")
        as_wrapper.terminate_instance(inst_id, True)
        logger.info("Terminated instance %s in group %s.", inst_id, group_name)
    print("Waiting for instances to stop...")
    wait_for_instances(instance_ids, as_wrapper)
    print(f"Deleting {group_name}.")
    as_wrapper.delete_autoscaling_group(group_name)
    logger.info("Deleted Auto Scaling group %s.", group_name)
    print("-" * 88)

    if template is not None:
        if q.ask(
            f"Do you want to delete launch template {template_name} used in this
demo (y/n)? "
        ):
            svc_helper.delete_template(template_name)
            logger.info("Deleted launch template %s.", template_name)
            print("Template deleted.")
```

```
    print("\nThanks for watching!")
    print("-" * 88)


if __name__ == "__main__":
    try:
        wrapper = AutoScalingWrapper(boto3.client("autoscaling"))
        helper = ServiceHelper(boto3.client("ec2"), boto3.resource("cloudwatch"))
        run_scenario(wrapper, helper)
    except Exception:
        logger.exception("Something went wrong with the demo!")
```

Definir las funciones a las que llama el escenario para administrar plantillas de lanzamiento y métricas. Estas funciones agrupan Amazon EC2 y CloudWatch sus acciones.

```
class ServiceHelper:
    """Encapsulates Amazon EC2 and CloudWatch actions for the example."""

    def __init__(self, ec2_client, cloudwatch_resource):
        """
        :param ec2_client: A Boto3 Amazon EC2 client.
        :param cloudwatch_resource: A Boto3 CloudWatch resource.
        """
        self.ec2_client = ec2_client
        self.cloudwatch_resource = cloudwatch_resource

    def get_template(self, template_name: str) -> dict:
        """
        Gets a launch template. Launch templates specify configuration for
 instances
        that are launched by Amazon EC2 Auto Scaling.

        :param template_name: The name of the template to look up.
        :return: The template, if it exists.
        :raises ClientError: If there is an error retrieving the launch template.
        """
        try:
            response = self.ec2_client.describe_launch_templates(
                LaunchTemplateNames=[template_name]
            )
            template = response["LaunchTemplates"][0]
```

```
            logger.info("Launch template %s retrieved successfully.",
template_name)
            return template
        except ClientError as err:
            if (
                err.response["Error"]["Code"]
                == "InvalidLaunchTemplateName.NotFoundException"
            ):
                logger.warning("Launch template %s does not exist.",
template_name)
            else:
                logger.error(
                    "Couldn't verify launch template %s. Error: %s: %s",
                    template_name,
                    err.response["Error"]["Code"],
                    err.response["Error"]["Message"],
                )
                raise

    def create_template(self, template_name: str, inst_type: str, ami_id: str) ->
dict:
        """
        Creates an Amazon EC2 launch template to use with Amazon EC2 Auto
Scaling.

        :param template_name: The name to give to the template.
        :param inst_type: The type of the instance, such as t1.micro.
        :param ami_id: The ID of the Amazon Machine Image (AMI) to use when
creating
                       an instance.
        :return: Information about the newly created template.
        :raises ClientError: If there is an error creating the launch template.
        """
        try:
            response = self.ec2_client.create_launch_template(
                LaunchTemplateName=template_name,
                LaunchTemplateData={"InstanceType": inst_type, "ImageId":
ami_id},
            )
            template = response["LaunchTemplate"]
            logger.info(
                "Created launch template %s with instance type %s and AMI ID
%s.",
                template_name,
```

```
                inst_type,
                ami_id,
            )
            return template
        except ClientError as err:
            logger.error(
                "Couldn't create launch template %s. Error: %s: %s",
                template_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise

    def delete_template(self, template_name: str) -> None:
        """
        Deletes a launch template.

        :param template_name: The name of the template to delete.
        :raises ClientError: If there is an error deleting the launch template.
        """
        try:

self.ec2_client.delete_launch_template(LaunchTemplateName=template_name)
            logger.info("Deleted launch template %s.", template_name)
        except ClientError as err:
            logger.error(
                "Couldn't delete launch template %s. Error: %s: %s",
                template_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise

    def get_availability_zones(self) -> list:
        """
        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.
        :raises ClientError: If there is an error retrieving availability zones.
        """
        try:
            response = self.ec2_client.describe_availability_zones()
            zones = [zone["ZoneName"] for zone in response["AvailabilityZones"]]
```

```
            logger.info("Retrieved availability zones: %s.", ", ".join(zones))
            return zones
        except ClientError as err:
            logger.error(
                "Couldn't get availability zones. Error: %s: %s",
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise

    def get_metrics(self, namespace: str, dimensions: list) -> list:
        """
        Gets a list of CloudWatch metrics filtered by namespace and dimensions.

        :param namespace: The namespace of the metrics to look up.
        :param dimensions: The dimensions of the metrics to look up.
        :return: The list of metrics.
        :raises ClientError: If there is an error retrieving CloudWatch metrics.
        """
        try:
            metrics = list(
                self.cloudwatch_resource.metrics.filter(
                    Namespace=namespace, Dimensions=dimensions
                )
            )
            logger.info(
                "Retrieved metrics for namespace %s with dimensions %s.",
                namespace,
                dimensions,
            )
            return metrics
        except ClientError as err:
            logger.error(
                "Couldn't get metrics for %s, %s. Error: %s: %s",
                namespace,
                dimensions,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise

    @staticmethod
    def get_metric_statistics(
        dimensions: list, metric, start: datetime, end: datetime
```

```
    ) -> list:
        """
        Gets statistics for a CloudWatch metric within a specified time span.

        :param dimensions: The dimensions of the metric.
        :param metric: The metric to look up.
        :param start: The start of the time span for retrieved metrics.
        :param end: The end of the time span for retrieved metrics.
        :return: The list of data points found for the specified metric.
        :raises ClientError: If there is an error retrieving metric statistics.
        """
        try:
            response = metric.get_statistics(
                Dimensions=dimensions,
                StartTime=start,
                EndTime=end,
                Period=60,
                Statistics=["Sum"],
            )
            data = response["Datapoints"]
            logger.info("Retrieved statistics for metric %s.", metric.name)
            return data
        except ClientError as err:
            logger.error(
                "Couldn't get statistics for metric %s. Error: %s: %s",
                metric.name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise


def print_simplified_group(group: dict) -> None:
    """
    Prints a subset of data for an Auto Scaling group.

    :param group: The Auto Scaling group data to print.
    :return: None
    """
    print(group["AutoScalingGroupName"])
    print(f"\tLaunch template: {group['LaunchTemplate']['LaunchTemplateName']}")
    print(
        f"\tMin: {group['MinSize']}, Max: {group['MaxSize']}, Desired:
 {group['DesiredCapacity']}"
```

```
    )
    if group["Instances"]:
        print(f"\tInstances:")
        for inst in group["Instances"]:
            print(f"\t\t{inst['InstanceId']}: {inst['LifecycleState']}")


def wait_for_group(group_name: str, as_wrapper: AutoScalingWrapper) -> list:
    """
    Waits for instances to start or stop in an Auto Scaling group.
    Prints the data for each instance after scaling activities are complete.

    :param group_name: The name of the Auto Scaling group.
    :param as_wrapper: The AutoScalingWrapper that manages Auto Scaling groups.
    :return: A list of instance IDs in the group.
    """
    group = as_wrapper.describe_group(group_name)
    instance_ids = [i["InstanceId"] for i in group["Instances"]]
    return wait_for_instances(instance_ids, as_wrapper)


def wait_for_instances(instance_ids: list, as_wrapper: AutoScalingWrapper) ->
 list:
    """
    Waits for instances to start or stop in an Auto Scaling group.
    Prints the data for each instance after scaling activities are complete.

    :param instance_ids: A list of instance IDs to wait for.
    :param as_wrapper: The AutoScalingWrapper that manages Auto Scaling groups.
    :return: A list of instance IDs that were waited on.
    """
    ready = False
    instances = []
    while not ready:
        instances = as_wrapper.describe_instances(instance_ids) if instance_ids
 else []
        if all([x["LifecycleState"] in ["Terminated", "InService"] for x in
 instances]):
            ready = True
        else:
            wait(10)
    if instances:
        print(
```

```
            f"Here are the details of the instance{'s' if len(instances) > 1 else
''}:"
        )
        for instance in instances:
            pp(instance)
    return instance_ids
```

- Para obtener información sobre la API, consulte los siguientes temas en la Referencia de la API de AWS SDK para Python (Boto3).
  - CreateAutoScalingGroup
  - DeleteAutoScalingGroup
  - DescribeAutoScalingGroups
  - DescribeAutoScalingInstances
  - DescribeScalingActivities
  - DisableMetricsCollection
  - EnableMetricsCollection
  - SetDesiredCapacity
  - TerminateInstanceInAutoScalingGroup
  - UpdateAutoScalingGroup

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.

```
[package]
name = "autoscaling-code-examples"
version = "0.1.0"
```

```
authors = ["Doug Schwartz <dougsch@amazon.com>", "David Souther
 <dpsouth@amazon.com>"]
edition = "2021"

# See more keys and their definitions at https://doc.rust-lang.org/cargo/
reference/manifest.html

[dependencies]
aws-config = { version = "1.0.1", features = ["behavior-version-latest"] }
aws-sdk-autoscaling = { version = "1.3.0" }
aws-sdk-ec2 = { version = "1.3.0" }
aws-types = { version = "1.0.1" }
tokio = { version = "1.20.1", features = ["full"] }
clap = { version = "4.4", features = ["derive"] }
tracing-subscriber = { version = "0.3.15", features = ["env-filter"] }
anyhow = "1.0.75"
tracing = "0.1.37"
tokio-stream = "0.1.14"


use std::{collections::BTreeSet, fmt::Display};

use anyhow::anyhow;
use autoscaling_code_examples::scenario::{AutoScalingScenario, ScenarioError};
use tracing::{info, warn};

async fn show_scenario_description(scenario: &AutoScalingScenario, event: &str) {
    let description = scenario.describe_scenario().await;
    info!("DescribeAutoScalingInstances: {event}\n{description}");
}

#[derive(Default, Debug)]
struct Warnings(Vec<String>);

impl Warnings {
    pub fn push(&mut self, warning: &str, error: ScenarioError) {
        let formatted = format!("{warning}: {error}");
        warn!("{formatted}");
        self.0.push(formatted);
    }

    pub fn is_empty(&self) -> bool {
        self.0.is_empty()
    }
```

```
}

impl Display for Warnings {
    fn fmt(&self, f: &mut std::fmt::Formatter<'_>) -> std::fmt::Result {
        writeln!(f, "Warnings:")?;
        for warning in &self.0 {
            writeln!(f, "{: >4}- {warning}", "")?;
        }
        Ok(())
    }
}

#[tokio::main]
async fn main() -> Result<(), anyhow::Error> {
    tracing_subscriber::fmt::init();

    let shared_config = aws_config::from_env().load().await;

    let mut warnings = Warnings::default();

    // 1. Create an EC2 launch template that you'll use to create an auto scaling
 group. Bonus: use SDK with EC2.CreateLaunchTemplate to create the launch
 template.
    // 2. CreateAutoScalingGroup: pass it the launch template you created in step
 0. Give it min/max of 1 instance.
    // 4. EnableMetricsCollection: enable all metrics or a subset.
    let scenario = match
 AutoScalingScenario::prepare_scenario(&shared_config).await {
        Ok(scenario) => scenario,
        Err(errs) => {
            let err_str = errs
                .into_iter()
                .map(|e| e.to_string())
                .collect::<Vec<String>>()
                .join(", ");
            return Err(anyhow!("Failed to initialize scenario: {err_str}"));
        }
    };

    info!("Prepared autoscaling scenario:\n{scenario}");

    let stable = scenario.wait_for_stable(1).await;
    if let Err(err) = stable {
        warnings.push(
```

```
            "There was a problem while waiting for group to be stable",
            err,
        );
    }

    // 3. DescribeAutoScalingInstances: show that one instance has launched.
    show_scenario_description(
        &scenario,
        "show that the group was created and one instance has launched",
    )
    .await;

    // 5. UpdateAutoScalingGroup: update max size to 3.
    let scale_max_size = scenario.scale_max_size(3).await;
    if let Err(err) = scale_max_size {
        warnings.push("There was a problem scaling max size", err);
    }

    // 6. DescribeAutoScalingGroups: the current state of the group
    show_scenario_description(
        &scenario,
        "show the current state of the group after setting max size",
    )
    .await;

    // 7. SetDesiredCapacity: set desired capacity to 2.
    let scale_desired_capacity = scenario.scale_desired_capacity(2).await;
    if let Err(err) = scale_desired_capacity {
        warnings.push("There was a problem setting desired capacity", err);
    }

    //   Wait for a second instance to launch.
    let stable = scenario.wait_for_stable(2).await;
    if let Err(err) = stable {
        warnings.push(
            "There was a problem while waiting for group to be stable",
            err,
        );
    }

    // 8. DescribeAutoScalingInstances: show that two instances are launched.
    show_scenario_description(
        &scenario,
        "show that two instances are launched after setting desired capacity",
```

```
    )
    .await;

    let ids_before = scenario
        .list_instances()
        .await
        .map(|v| v.into_iter().collect::<BTreeSet<_>>())
        .unwrap_or_default();

    // 9. TerminateInstanceInAutoScalingGroup: terminate one of the instances in
the group.
    let terminate_some_instance = scenario.terminate_some_instance().await;
    if let Err(err) = terminate_some_instance {
        warnings.push("There was a problem replacing an instance", err);
    }

    let wait_after_terminate = scenario.wait_for_stable(1).await;
    if let Err(err) = wait_after_terminate {
        warnings.push(
            "There was a problem waiting after terminating an instance",
            err,
        );
    }

    let wait_scale_up_after_terminate = scenario.wait_for_stable(2).await;
    if let Err(err) = wait_scale_up_after_terminate {
        warnings.push(
            "There was a problem waiting for scale up after terminating an
instance",
            err,
        );
    }

    let ids_after = scenario
        .list_instances()
        .await
        .map(|v| v.into_iter().collect::<BTreeSet<_>>())
        .unwrap_or_default();

    let difference = ids_after.intersection(&ids_before).count();
    if !(difference == 1 && ids_before.len() == 2 && ids_after.len() == 2) {
        warnings.push(
            "Before and after set not different",
            ScenarioError::with(format!("{difference}")),
```

```
        );
    }

    // 10. DescribeScalingActivities: list the scaling activities that have
 occurred for the group so far.
    show_scenario_description(
        &scenario,
        "list the scaling activities that have occurred for the group so far",
    )
    .await;

    // 11. DisableMetricsCollection
    let scale_group = scenario.scale_group_to_zero().await;
    if let Err(err) = scale_group {
        warnings.push("There was a problem scaling the group to 0", err);
    }
    show_scenario_description(&scenario, "Scenario scaled to 0").await;

    // 12. DeleteAutoScalingGroup (to delete the group you must stop all
 instances):
    // 13. Delete LaunchTemplate.
    let clean_scenario = scenario.clean_scenario().await;
    if let Err(errs) = clean_scenario {
        for err in errs {
            warnings.push("There was a problem cleaning the scenario", err);
        }
    } else {
        info!("The scenario has been cleaned up!");
    }

    if warnings.is_empty() {
        Ok(())
    } else {
        Err(anyhow!(
            "There were warnings during scenario execution:\n{warnings}"
        ))
    }
}

pub mod scenario;


use std::{
    error::Error,
```

```
    fmt::{Debug, Display},
    time::{Duration, SystemTime},
};

use anyhow::anyhow;
use aws_config::SdkConfig;
use aws_sdk_autoscaling::{
    error::{DisplayErrorContext, ProvideErrorMetadata},
    types::{Activity, AutoScalingGroup, LaunchTemplateSpecification},
};
use aws_sdk_ec2::types::RequestLaunchTemplateData;
use tracing::trace;

const LAUNCH_TEMPLATE_NAME: &str =
 "SDK_Code_Examples_EC2_Autoscaling_template_from_Rust_SDK";
const AUTOSCALING_GROUP_NAME: &str =
 "SDK_Code_Examples_EC2_Autoscaling_Group_from_Rust_SDK";
const MAX_WAIT: Duration = Duration::from_secs(5 * 60); // Wait at most 25
 seconds.
const WAIT_TIME: Duration = Duration::from_millis(500); // Wait half a second at
 a time.

struct Waiter {
    start: SystemTime,
    max: Duration,
}

impl Waiter {
    fn new() -> Self {
        Waiter {
            start: SystemTime::now(),
            max: MAX_WAIT,
        }
    }

    async fn sleep(&self) -> Result<(), ScenarioError> {
        if SystemTime::now()
            .duration_since(self.start)
            .unwrap_or(Duration::MAX)
            > self.max
        {
            Err(ScenarioError::with(
                "Exceeded maximum wait duration for stable group",
            ))
```

```
        } else {
            tokio::time::sleep(WAIT_TIME).await;
            Ok(())
        }
    }
}

pub struct AutoScalingScenario {
    ec2: aws_sdk_ec2::Client,
    autoscaling: aws_sdk_autoscaling::Client,
    launch_template_arn: String,
    auto_scaling_group_name: String,
}

impl Display for AutoScalingScenario {
    fn fmt(&self, f: &mut std::fmt::Formatter<'_>) -> std::fmt::Result {
        f.write_fmt(format_args!(
            "\tLaunch Template ID: {}\n",
            self.launch_template_arn
        ))?;
        f.write_fmt(format_args!(
            "\tScaling Group Name: {}\n",
            self.auto_scaling_group_name
        ))?;

        Ok(())
    }
}

pub struct AutoScalingScenarioDescription {
    group: Result<Vec<String>, ScenarioError>,
    instances: Result<Vec<String>, anyhow::Error>,
    activities: Result<Vec<Activity>, anyhow::Error>,
}

impl Display for AutoScalingScenarioDescription {
    fn fmt(&self, f: &mut std::fmt::Formatter<'_>) -> std::fmt::Result {
        writeln!(f, "\t      Group status:")?;
        match &self.group {
            Ok(groups) => {
                for status in groups {
                    writeln!(f, "\t\t- {status}")?;
                }
            }
```

```
            Err(e) => writeln!(f, "\t\t! - {e}")?,
        }
        writeln!(f, "\t         Instances:")?;
        match &self.instances {
            Ok(instances) => {
                for instance in instances {
                    writeln!(f, "\t\t- {instance}")?;
                }
            }
            Err(e) => writeln!(f, "\t\t! {e}")?,
        }

        writeln!(f, "\t        Activities:")?;
        match &self.activities {
            Ok(activities) => {
                for activity in activities {
                    writeln!(
                        f,
                        "\t\t- {} Progress: {}% Status: {:?} End: {:?}",
                        activity.cause().unwrap_or("Unknown"),
                        activity.progress.unwrap_or(-1),
                        activity.status_code(),
                        // activity.status_message().unwrap_or_default()
                        activity.end_time(),
                    )?;
                }
            }
            Err(e) => writeln!(f, "\t\t! {e}")?,
        }

        Ok(())
    }
}

#[derive(Debug)]
struct MetadataError {
    message: Option<String>,
    code: Option<String>,
}

impl MetadataError {
    fn from(err: &dyn ProvideErrorMetadata) -> Self {
        MetadataError {
            message: err.message().map(|s| s.to_string()),
```

```
            code: err.code().map(|s| s.to_string()),
        }
    }
}

impl Display for MetadataError {
    fn fmt(&self, f: &mut std::fmt::Formatter<'_>) -> std::fmt::Result {
        let display = match (&self.message, &self.code) {
            (None, None) => "Unknown".to_string(),
            (None, Some(code)) => format!("({code})"),
            (Some(message), None) => message.to_string(),
            (Some(message), Some(code)) => format!("{message} ({code})"),
        };
        write!(f, "{display}")
    }
}

#[derive(Debug)]
pub struct ScenarioError {
    message: String,
    context: Option<MetadataError>,
}

impl ScenarioError {
    pub fn with(message: impl Into<String>) -> Self {
        ScenarioError {
            message: message.into(),
            context: None,
        }
    }

    pub fn new(message: impl Into<String>, err: &dyn ProvideErrorMetadata) ->
 Self {
        ScenarioError {
            message: message.into(),
            context: Some(MetadataError::from(err)),
        }
    }
}

impl Error for ScenarioError {
    // While `Error` can capture `source` information about the underlying error,
 for this example
```

```
    // the ScenarioError captures the underlying information in MetadataError and
 treats it as a
    // single Error from this Crate. In other contexts, it may be appropriate to
 model the error
    // as including the SdkError as its source.
}
impl Display for ScenarioError {
    fn fmt(&self, f: &mut std::fmt::Formatter<'_>) -> std::fmt::Result {
        match &self.context {
            Some(c) => write!(f, "{}: {}", self.message, c),
            None => write!(f, "{}", self.message),
        }
    }
}

impl AutoScalingScenario {
    pub async fn prepare_scenario(sdk_config: &SdkConfig) -> Result<Self,
 Vec<ScenarioError>> {
        let ec2 = aws_sdk_ec2::Client::new(sdk_config);
        let autoscaling = aws_sdk_autoscaling::Client::new(sdk_config);

        let auto_scaling_group_name = String::from(AUTOSCALING_GROUP_NAME);

        // Before creating any resources, prepare the list of AZs
        let availablity_zones = ec2.describe_availability_zones().send().await;
        if let Err(err) = availablity_zones {
            return Err(vec![ScenarioError::new("Failed to find AZs", &err)]);
        }

        let availability_zones: Vec<String> = availablity_zones
            .unwrap()
            .availability_zones
            .unwrap_or_default()
            .iter()
            .take(3)
            .map(|z| z.zone_name.clone().unwrap())
            .collect();

        // 1. Create an EC2 launch template that you'll use to create an auto
 scaling group. Bonus: use SDK with EC2.CreateLaunchTemplate to create the launch
 template.
        //   * Recommended: InstanceType='t1.micro',
 ImageId='ami-0ca285d4c2cda3300'
        let create_launch_template = ec2
```

```
            .create_launch_template()
            .launch_template_name(LAUNCH_TEMPLATE_NAME)
            .launch_template_data(
                RequestLaunchTemplateData::builder()
                    .instance_type(aws_sdk_ec2::types::InstanceType::T1Micro)
                    .image_id("ami-0ca285d4c2cda3300")
                    .build(),
            )
            .send()
            .await
            .map_err(|err| vec![ScenarioError::new("Failed to create launch
template", &err)])?;

        let launch_template_arn = match create_launch_template.launch_template {
            Some(launch_template) =>
launch_template.launch_template_id.unwrap_or_default(),
            None => {
                // Try to delete the launch template
                let _ = ec2
                    .delete_launch_template()
                    .launch_template_name(LAUNCH_TEMPLATE_NAME)
                    .send()
                    .await;
                return Err(vec![ScenarioError::with("Failed to load launch
template")]);
            }
        };

        // 2. CreateAutoScalingGroup: pass it the launch template you created in
step 0. Give it min/max of 1 instance.
        //   You can use EC2.describe_availability_zones() to get a list of AZs
(you have to specify an AZ when you create the group).
        //   Wait for instance to launch. Use a waiter if you have one, otherwise
DescribeAutoScalingInstances until LifecycleState='InService'
        if let Err(err) = autoscaling
            .create_auto_scaling_group()
            .auto_scaling_group_name(auto_scaling_group_name.as_str())
            .launch_template(
                LaunchTemplateSpecification::builder()
                    .launch_template_id(launch_template_arn.clone())
                    .version("$Latest")
                    .build(),
            )
            .max_size(1)
```

```
            .min_size(1)
            .set_availability_zones(Some(availability_zones))
            .send()
            .await
        {
            let mut errs = vec![ScenarioError::new(
                "Failed to create autoscaling group",
                &err,
            )];

            if let Err(err) = autoscaling
                .delete_auto_scaling_group()
                .auto_scaling_group_name(auto_scaling_group_name.as_str())
                .send()
                .await
            {
                errs.push(ScenarioError::new(
                    "Failed to clean up autoscaling group",
                    &err,
                ));
            }

            if let Err(err) = ec2
                .delete_launch_template()
                .launch_template_id(launch_template_arn.clone())
                .send()
                .await
            {
                errs.push(ScenarioError::new(
                    "Failed to clean up launch template",
                    &err,
                ));
            }
            return Err(errs);
        }

        let scenario = AutoScalingScenario {
            ec2,
            autoscaling: autoscaling.clone(), // Clients are cheap so cloning
here to prevent a move is ok.
            auto_scaling_group_name: auto_scaling_group_name.clone(),
            launch_template_arn,
        };
```

```
        let enable_metrics_collection = autoscaling
            .enable_metrics_collection()
            .auto_scaling_group_name(auto_scaling_group_name.as_str())
            .granularity("1Minute")
            .set_metrics(Some(vec![
                String::from("GroupMinSize"),
                String::from("GroupMaxSize"),
                String::from("GroupDesiredCapacity"),
                String::from("GroupInServiceInstances"),
                String::from("GroupTotalInstances"),
            ]))
            .send()
            .await;

        match enable_metrics_collection {
            Ok(_) => Ok(scenario),
            Err(err) => {
                scenario.clean_scenario().await?;
                Err(vec![ScenarioError::new(
                    "Failed to enable metrics collections for group",
                    &err,
                )])
            }
        }
    }

    pub async fn clean_scenario(self) -> Result<(), Vec<ScenarioError>> {
        let _ = self.wait_for_no_scaling().await;
        let delete_group = self
            .autoscaling
            .delete_auto_scaling_group()
            .auto_scaling_group_name(self.auto_scaling_group_name.clone())
            .send()
            .await;

        // 14. Delete LaunchTemplate.
        let delete_launch_template = self
            .ec2
            .delete_launch_template()
            .launch_template_id(self.launch_template_arn.clone())
            .send()
            .await;

        let early_exit = match (delete_group, delete_launch_template) {
```

```
            (Ok(_), Ok(_)) => Ok(()),
            (Ok(_), Err(e)) => Err(vec![ScenarioError::new(
                "There was an error cleaning the launch template",
                &e,
            )]),
            (Err(e), Ok(_)) => Err(vec![ScenarioError::new(
                "There was an error cleaning the scale group",
                &e,
            )]),
            (Err(e1), Err(e2)) => Err(vec![
                ScenarioError::new("Multiple error cleaning the scenario Scale
Group", &e1),
                ScenarioError::new("Multiple error cleaning the scenario Launch
Template", &e2),
            ]),
        };

        if early_exit.is_err() {
            early_exit
        } else {
            // Wait for delete_group to finish
            let waiter = Waiter::new();
            let mut errors = Vec::<ScenarioError>::new();
            while errors.len() < 3 {
                if let Err(e) = waiter.sleep().await {
                    errors.push(e);
                    continue;
                }
                let describe_group = self
                    .autoscaling
                    .describe_auto_scaling_groups()

.auto_scaling_group_names(self.auto_scaling_group_name.clone())
                    .send()
                    .await;
                match describe_group {
                    Ok(group) => match group.auto_scaling_groups().first() {
                        Some(group) => {
                            if group.status() != Some("Delete in progress") {
                                errors.push(ScenarioError::with(format!(
                                    "Group in an unknown state while deleting:
{}",
                                    group.status().unwrap_or("unknown error")
                                )));
```

```
                            return Err(errors);
                        }
                    }
                    None => return Ok(()),
                },
                Err(err) => {
                    errors.push(ScenarioError::new("Failed to describe
autoscaling group during cleanup 3 times, last error", &err));
                }
            }
            if errors.len() > 3 {
                return Err(errors);
            }
        }
        Err(vec![ScenarioError::with(
            "Exited cleanup wait loop without retuning success or failing
after three rounds",
        )])
    }
}

pub async fn describe_scenario(&self) -> AutoScalingScenarioDescription {
    let group = self
        .autoscaling
        .describe_auto_scaling_groups()
        .auto_scaling_group_names(self.auto_scaling_group_name.clone())
        .send()
        .await
        .map(|s| {
            s.auto_scaling_groups()
                .iter()
                .map(|s| {
                    format!(
                        "{}: {}",
                        s.auto_scaling_group_name().unwrap_or("Unknown"),
                        s.status().unwrap_or("Unknown")
                    )
                })
                .collect::<Vec<String>>()
        })
        .map_err(|e| {
            ScenarioError::new("Failed to describe auto scaling groups for
scenario", &e)
        });
```

```
        let instances = self
            .list_instances()
            .await
            .map_err(|e| anyhow!("There was an error listing instances: {e}",));

        // 10. DescribeScalingActivities: list the scaling activities that have
occurred for the group so far.
        //   Bonus: use CloudWatch API to get and show some metrics collected for
the group.
        //   CW.ListMetrics with Namespace='AWS/AutoScaling' and
Dimensions=[{'Name': 'AutoScalingGroupName', 'Value': }]
        //   CW.GetMetricStatistics with Statistics='Sum'. Start and End times
must be in UTC!
        let activities = self
            .autoscaling
            .describe_scaling_activities()
            .auto_scaling_group_name(self.auto_scaling_group_name.clone())
            .into_paginator()
            .items()
            .send()
            .collect::<Result<Vec<_>, _>>()
            .await
            .map_err(|e| {
                anyhow!(
                    "There was an error retrieving scaling activities: {}",
                    DisplayErrorContext(&e)
                )
            });

        AutoScalingScenarioDescription {
            group,
            instances,
            activities,
        }
    }

    async fn get_group(&self) -> Result<AutoScalingGroup, ScenarioError> {
        let describe_auto_scaling_groups = self
            .autoscaling
            .describe_auto_scaling_groups()
            .auto_scaling_group_names(self.auto_scaling_group_name.clone())
            .send()
            .await;
```

```
        if let Err(err) = describe_auto_scaling_groups {
            return Err(ScenarioError::new(
                format!(
                    "Failed to get status of autoscaling group {}",
                    self.auto_scaling_group_name.clone()
                )
                .as_str(),
                &err,
            ));
        }

        let describe_auto_scaling_groups_output =
describe_auto_scaling_groups.unwrap();
        let auto_scaling_groups =
describe_auto_scaling_groups_output.auto_scaling_groups();
        let auto_scaling_group = auto_scaling_groups.first();

        if auto_scaling_group.is_none() {
            return Err(ScenarioError::with(format!(
                "Could not find autoscaling group {}",
                self.auto_scaling_group_name.clone()
            )));
        }

        Ok(auto_scaling_group.unwrap().clone())
    }

    pub async fn wait_for_no_scaling(&self) -> Result<(), ScenarioError> {
        let waiter = Waiter::new();
        let mut scaling = true;
        while scaling {
            waiter.sleep().await?;
            let describe_activities = self
                .autoscaling
                .describe_scaling_activities()
                .auto_scaling_group_name(self.auto_scaling_group_name.clone())
                .send()
                .await
                .map_err(|e| {
                    ScenarioError::new("Failed to get autoscaling activities for
group", &e)
                })?;
            let activities = describe_activities.activities();
```

```
            trace!(
                "Waiting for no scaling found {} activities",
                activities.len()
            );
            scaling = activities.iter().any(|a| a.progress() < Some(100));
        }
        Ok(())
    }

    pub async fn wait_for_stable(&self, size: usize) -> Result<(), ScenarioError>
 {
        self.wait_for_no_scaling().await?;

        let mut group = self.get_group().await?;
        let mut count = count_group_instances(&group);

        let waiter = Waiter::new();
        while count != size {
            trace!("Waiting for stable {size} (current: {count})");
            waiter.sleep().await?;
            group = self.get_group().await?;
            count = count_group_instances(&group);
        }

        Ok(())
    }

    pub async fn list_instances(&self) -> Result<Vec<String>, ScenarioError> {
        // The direct way to list instances is by using
 DescribeAutoScalingGroup's instances property. However, this returns a
 Vec<Instance>, as opposed to a Vec<AutoScalingInstanceDetails>.
        // Ok(self.get_group().await?.instances.unwrap_or_default().map(|
i| i.instance_id.clone().unwrap_or_default()).filter(|id| !
id.is_empty()).collect())

        // Alternatively, and for the sake of example,
 DescribeAutoScalingInstances returns a list that can be filtered by the client.
        self.autoscaling
            .describe_auto_scaling_instances()
            .into_paginator()
            .items()
            .send()
            .try_collect()
            .await
```

```
            .map(|items| {
                items
                    .into_iter()
                    .filter(|i| {
                        i.auto_scaling_group_name.as_deref()
                            == Some(self.auto_scaling_group_name.as_str())
                    })
                    .map(|i| i.instance_id.unwrap_or_default())
                    .filter(|id| !id.is_empty())
                    .collect::<Vec<String>>()
            })
            .map_err(|err| ScenarioError::new("Failed to get list of auto scaling
instances", &err))
    }

    pub async fn scale_min_size(&self, size: i32) -> Result<(), ScenarioError> {
        let update_group = self
            .autoscaling
            .update_auto_scaling_group()
            .auto_scaling_group_name(self.auto_scaling_group_name.clone())
            .min_size(size)
            .send()
            .await;
        if let Err(err) = update_group {
            return Err(ScenarioError::new(
                format!("Failer to update group to min size ({size}))").as_str(),
                &err,
            ));
        }
        Ok(())
    }

    pub async fn scale_max_size(&self, size: i32) -> Result<(), ScenarioError> {
        // 5. UpdateAutoScalingGroup: update max size to 3.
        let update_group = self
            .autoscaling
            .update_auto_scaling_group()
            .auto_scaling_group_name(self.auto_scaling_group_name.clone())
            .max_size(size)
            .send()
            .await;
        if let Err(err) = update_group {
            return Err(ScenarioError::new(
                format!("Failed to update group to max size ({size})").as_str(),
```

```
                &err,
            ));
        }
        Ok(())
    }

    pub async fn scale_desired_capacity(&self, capacity: i32) -> Result<(),
ScenarioError> {
        // 7. SetDesiredCapacity: set desired capacity to 2.
        //   Wait for a second instance to launch.
        let update_group = self
            .autoscaling
            .set_desired_capacity()
            .auto_scaling_group_name(self.auto_scaling_group_name.clone())
            .desired_capacity(capacity)
            .send()
            .await;
        if let Err(err) = update_group {
            return Err(ScenarioError::new(
                format!("Failed to update group to desired capacity
({capacity}))").as_str(),
                &err,
            ));
        }
        Ok(())
    }

    pub async fn scale_group_to_zero(&self) -> Result<(), ScenarioError> {
        // If this fails it's fine, just means there are extra cloudwatch metrics
events for the scale-down.
        let _ = self
            .autoscaling
            .disable_metrics_collection()
            .auto_scaling_group_name(self.auto_scaling_group_name.clone())
            .send()
            .await;

        // 12. DeleteAutoScalingGroup (to delete the group you must stop all
instances):
        //   UpdateAutoScalingGroup with MinSize=0
        let update_group = self
            .autoscaling
            .update_auto_scaling_group()
            .auto_scaling_group_name(self.auto_scaling_group_name.clone())
```

```
            .min_size(0)
            .desired_capacity(0)
            .send()
            .await;
        if let Err(err) = update_group {
            return Err(ScenarioError::new(
                "Failed to update group for scaling down&",
                &err,
            ));
        }

        let stable = self.wait_for_stable(0).await;
        if let Err(err) = stable {
            return Err(ScenarioError::with(format!(
                "Error while waiting for group to be stable on scale down: {err}"
            )));
        }

        Ok(())
    }

    pub async fn terminate_some_instance(&self) -> Result<(), ScenarioError> {
        // Retrieve a list of instances in the auto scaling group.
        let auto_scaling_group = self.get_group().await?;
        let instances = auto_scaling_group.instances();
        // Or use other logic to find an instance to terminate.
        let instance = instances.first();
        if let Some(instance) = instance {
            let instance_id = if let Some(instance_id) = instance.instance_id() {
                instance_id
            } else {
                return Err(ScenarioError::with("Missing instance id"));
            };
            let termination = self
                .ec2
                .terminate_instances()
                .instance_ids(instance_id)
                .send()
                .await;
            if let Err(err) = termination {
                Err(ScenarioError::new(
                    "There was a problem terminating an instance",
                    &err,
                ))
```

```
            } else {
                Ok(())
            }
        } else {
            Err(ScenarioError::with("There was no instance to terminate"))
        }
    }
}

fn count_group_instances(group: &AutoScalingGroup) -> usize {
    group.instances.as_ref().map(|i| i.len()).unwrap_or(0)
}
```

- Para obtener información sobre la API, consulte los siguientes temas en la Referencia de la API de AWS SDK para Rust.
  - CreateAutoScalingGroup
  - DeleteAutoScalingGroup
  - DescribeAutoScalingGroups
  - DescribeAutoScalingInstances
  - DescribeScalingActivities
  - DisableMetricsCollection
  - EnableMetricsCollection
  - SetDesiredCapacity
  - TerminateInstanceInAutoScalingGroup
  - UpdateAutoScalingGroup

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulteUtilización de este servicio con un AWS SDK. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Acciones para Auto Scaling usando AWS SDKs

Los siguientes ejemplos de código muestran cómo realizar acciones individuales de Auto Scaling con AWS SDKs. Cada ejemplo incluye un enlace a GitHub, donde puede encontrar instrucciones para configurar y ejecutar el código.

Estos fragmentos llaman a la API de escalado automático y son fragmentos de código de programas más grandes que deben ejecutarse en contexto. Puede ver las acciones en contexto en Escenarios para Auto Scaling usando AWS SDKs.

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 EC2 Auto Scaling.

Ejemplos

- Utilizar AttachInstances con una CLI
- Úselo AttachLoadBalancerTargetGroups con un AWS SDK o CLI
- Utilizar AttachLoadBalancers con una CLI
- Utilizar CompleteLifecycleAction con una CLI
- Úselo CreateAutoScalingGroup con un AWS SDK o CLI
- Utilizar CreateLaunchConfiguration con una CLI
- Utilizar CreateOrUpdateTags con una CLI
- Úselo DeleteAutoScalingGroup con un AWS SDK o CLI
- Utilizar DeleteLaunchConfiguration con una CLI
- Utilizar DeleteLifecycleHook con una CLI
- Utilizar DeleteNotificationConfiguration con una CLI
- Utilizar DeletePolicy con una CLI
- Utilizar DeleteScheduledAction con una CLI
- Utilizar DeleteTags con una CLI
- Utilizar DescribeAccountLimits con una CLI
- Utilizar DescribeAdjustmentTypes con una CLI
- Úselo DescribeAutoScalingGroups con un AWS SDK o CLI
- Úselo DescribeAutoScalingInstances con un AWS SDK o CLI
- Utilizar DescribeAutoScalingNotificationTypes con una CLI
- Utilizar DescribeLaunchConfigurations con una CLI
- Utilizar DescribeLifecycleHookTypes con una CLI
- Utilizar DescribeLifecycleHooks con una CLI
- Utilizar DescribeLoadBalancers con una CLI
- Utilizar DescribeMetricCollectionTypes con una CLI

- Utilizar DescribeNotificationConfigurations con una CLI
- Utilizar DescribePolicies con una CLI
- Úselo DescribeScalingActivities con un AWS SDK o CLI
- Utilizar DescribeScalingProcessTypes con una CLI
- Utilizar DescribeScheduledActions con una CLI
- Utilizar DescribeTags con una CLI
- Utilizar DescribeTerminationPolicyTypes con una CLI
- Utilizar DetachInstances con una CLI
- Utilizar DetachLoadBalancers con una CLI
- Úselo DisableMetricsCollection con un AWS SDK o CLI
- Úselo EnableMetricsCollection con un AWS SDK o CLI
- Utilizar EnterStandby con una CLI
- Utilizar ExecutePolicy con una CLI
- Utilizar ExitStandby con una CLI
- Utilizar PutLifecycleHook con una CLI
- Utilizar PutNotificationConfiguration con una CLI
- Utilizar PutScalingPolicy con una CLI
- Utilizar PutScheduledUpdateGroupAction con una CLI
- Utilizar RecordLifecycleActionHeartbeat con una CLI
- Utilizar ResumeProcesses con una CLI
- Úselo SetDesiredCapacity con un AWS SDK o CLI
- Utilizar SetInstanceHealth con una CLI
- Utilizar SetInstanceProtection con una CLI
- Utilizar SuspendProcesses con una CLI
- Úselo TerminateInstanceInAutoScalingGroup con un AWS SDK o CLI
- Úselo UpdateAutoScalingGroup con un AWS SDK o CLI

Utilizar **`AttachInstances`** con una CLI

Los siguientes ejemplos de código muestran cómo utilizar `AttachInstances`.

CLI

AWS CLI

Cómo asociar una instancia a un grupo de escalado automático

En este ejemplo, se asocia la instancia especificada al grupo de escalado automático especificado.

```
aws autoscaling attach-instances \
    --instance-ids i-061c63c5eb45f0416 \
    --auto-scaling-group-name my-asg
```

Este comando no genera ninguna salida.

- Para obtener más información sobre la API, consulte AttachInstancesla Referencia de AWS CLI comandos.

PowerShell

Herramientas para la PowerShell versión 4

Ejemplo 1: En este ejemplo, se asocia la instancia especificada al grupo de escalado automático especificado. El escalado automático aumenta automáticamente la capacidad deseada del grupo de escalado automático.

```
Mount-ASInstance -InstanceId i-93633f9b -AutoScalingGroupName my-asg
```

- Para obtener más información sobre la API, consulte AttachInstances Herramientas de AWS para PowerShellCmdlet Reference (V4).

Herramientas para la versión 5 PowerShell

Ejemplo 1: En este ejemplo, se asocia la instancia especificada al grupo de escalado automático especificado. El escalado automático aumenta automáticamente la capacidad deseada del grupo de escalado automático.

```
Mount-ASInstance -InstanceId i-93633f9b -AutoScalingGroupName my-asg
```

- Para obtener más información sobre la API, consulte AttachInstancesla referencia de Herramientas de AWS para PowerShell cmdlets (V5).

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. Utilización de este servicio con un AWS SDK En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Úselo **`AttachLoadBalancerTargetGroups`** con un AWS SDK o CLI

Los siguientes ejemplos de código muestran cómo utilizar `AttachLoadBalancerTargetGroups`.

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

SDK para .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>
    /// 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 }
            });
```

```
    }
```

- Para obtener más información sobre la API, consulta AttachLoadBalancerTargetGroupsla Referencia AWS SDK para .NET de la API.

CLI

AWS CLI

Asociación de un grupo de destino a un grupo de escalado automático

En este ejemplo, se asocia el grupo de destino especificado al grupo de escalado automático especificado.

```
aws autoscaling attach-load-balancer-target-groups \
    --auto-scaling-group-name my-asg \
    --target-group-arns arn:aws:elasticloadbalancing:us-
west-2:123456789012:targetgroup/my-targets/73e2d6bc24d8a067
```

Este comando no genera ninguna salida.

Para obtener más información, consulte Elastic Load Balancing y Amazon EC2 Auto Scaling en la Guía del usuario de Amazon EC2 Auto Scaling.

- Para obtener más información sobre la API, consulta AttachLoadBalancerTargetGroupsla 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 AutoScalingClient({});
    await client.send(
```

```
      new AttachLoadBalancerTargetGroupsCommand({
        AutoScalingGroupName: NAMES.autoScalingGroupName,
        TargetGroupARNs: [state.targetGroupArn],
      }),
    );
```

- Para obtener más información sobre la API, consulta AttachLoadBalancerTargetGroupsla Referencia AWS SDK para 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 AutoScalingWrapper:
    """
    Encapsulates Amazon EC2 Auto Scaling and EC2 management actions.
    """

    def __init__(
        self,
        resource_prefix: str,
        inst_type: str,
        ami_param: str,
        autoscaling_client: boto3.client,
        ec2_client: boto3.client,
        ssm_client: boto3.client,
        iam_client: boto3.client,
    ):
        """
        Initializes the AutoScaler class with the necessary parameters.

        :param resource_prefix: The prefix for naming AWS resources that are
 created by this class.
        :param inst_type: The type of EC2 instance to create, such as t3.micro.
```

```
        :param ami_param: The Systems Manager parameter used to look up the AMI
that is created.
        :param autoscaling_client: A Boto3 EC2 Auto Scaling client.
        :param ec2_client: A Boto3 EC2 client.
        :param ssm_client: A Boto3 Systems Manager client.
        :param iam_client: A Boto3 IAM client.
        """
        self.inst_type = inst_type
        self.ami_param = ami_param
        self.autoscaling_client = autoscaling_client
        self.ec2_client = ec2_client
        self.ssm_client = ssm_client
        self.iam_client = iam_client
        sts_client = boto3.client("sts")
        self.account_id = sts_client.get_caller_identity()["Account"]

        self.key_pair_name = f"{resource_prefix}-key-pair"
        self.launch_template_name = f"{resource_prefix}-template-"
        self.group_name = f"{resource_prefix}-group"

        # Happy path
        self.instance_policy_name = f"{resource_prefix}-pol"
        self.instance_role_name = f"{resource_prefix}-role"
        self.instance_profile_name = f"{resource_prefix}-prof"

        # Failure mode
        self.bad_creds_policy_name = f"{resource_prefix}-bc-pol"
        self.bad_creds_role_name = f"{resource_prefix}-bc-role"
        self.bad_creds_profile_name = f"{resource_prefix}-bc-prof"


    def attach_load_balancer_target_group(
        self, lb_target_group: Dict[str, Any]
    ) -> None:
        """
        Attaches an Elastic Load Balancing (ELB) target group to this EC2 Auto
Scaling group.
        The target group specifies how the load balancer forwards requests to the
instances
        in the group.

        :param lb_target_group: Data about the ELB target group to attach.
        """
        try:
```

```
            self.autoscaling_client.attach_load_balancer_target_groups(
                AutoScalingGroupName=self.group_name,
                TargetGroupARNs=[lb_target_group["TargetGroupArn"]],
            )
            log.info(
                "Attached load balancer target group %s to auto scaling group
%s.",
                lb_target_group["TargetGroupName"],
                self.group_name,
            )
        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            log.error(
                f"Failed to attach load balancer target group
'{lb_target_group['TargetGroupName']}'."
            )
            if error_code == "ResourceContentionFault":
                log.error(
                    "The request failed due to a resource contention issue. "
                    "Ensure that no conflicting operations are being performed on
the resource."
                )
            elif error_code == "ServiceLinkedRoleFailure":
                log.error(
                    "The operation failed because the service-linked role is not
ready or does not exist. "
                    "Check that the service-linked role exists and is correctly
configured."
                )
            log.error(f"Full error:\n\t{err}")
```

- Para obtener más información sobre la API, consulta AttachLoadBalancerTargetGroupsla 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. Utilización de este servicio con un AWS SDK En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Utilizar **`AttachLoadBalancers`** con una CLI

Los siguientes ejemplos de código muestran cómo utilizar `AttachLoadBalancers`.

CLI

AWS CLI

Cómo asociar un equilibrador de carga clásico al grupo de escalado automático

En este ejemplo, se asocia el equilibrador de carga clásico especificado al grupo de escalado automático especificado.

```
aws autoscaling attach-load-balancers \
    --load-balancer-names my-load-balancer \
    --auto-scaling-group-name my-asg
```

Este comando no genera ninguna salida.

Para obtener más información, consulte Elastic Load Balancing y Amazon EC2 Auto Scaling en la Guía del usuario de Amazon EC2 Auto Scaling.

- Para obtener más información sobre la API, consulte AttachLoadBalancersla Referencia de AWS CLI comandos.

PowerShell

Herramientas para la PowerShell versión 4

Ejemplo 1: En este ejemplo se asocia el equilibrador de carga especificado al grupo de escalado automático especificado.

```
Add-ASLoadBalancer -LoadBalancerName my-lb -AutoScalingGroupName my-asg
```

- Para obtener más información sobre la API, consulte AttachLoadBalancers Herramientas de AWS para PowerShellCmdlet Reference (V4).

Herramientas para la versión 5 PowerShell

Ejemplo 1: En este ejemplo se asocia el equilibrador de carga especificado al grupo de escalado automático especificado.

```
Add-ASLoadBalancer -LoadBalancerName my-lb -AutoScalingGroupName my-asg
```

- Para obtener más información sobre la API, consulte AttachLoadBalancersla referencia de Herramientas de AWS para PowerShell cmdlets (V5).

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. Utilización de este servicio con un AWS SDK En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Utilizar **`CompleteLifecycleAction`** con una CLI

Los siguientes ejemplos de código muestran cómo utilizar `CompleteLifecycleAction`.

CLI

AWS CLI

Cómo completar una acción del ciclo de vida

En este ejemplo, se notifica a Amazon EC2 Auto Scaling que la acción del ciclo de vida especificada se ha completado para que pueda finalizar la inicialización o la finalización de la instancia.

```
aws autoscaling complete-lifecycle-action \
    --lifecycle-hook-name my-launch-hook \
    --auto-scaling-group-name my-asg \
    --lifecycle-action-result CONTINUE \
    --lifecycle-action-token bcd2f1b8-9a78-44d3-8a7a-4dd07d7cf635
```

Este comando no genera ninguna salida.

Para obtener más información, consulte Enlaces de ciclo de vida de Amazon EC2 Auto Scaling en la guía del usuario de Amazon EC2 Auto Scaling.

- Para obtener más información sobre la API, consulte CompleteLifecycleActionla Referencia de AWS CLI comandos.

PowerShell

Herramientas para la PowerShell versión 4

Ejemplo 1: En este ejemplo se completa la acción del ciclo de vida especificada.

```
Complete-ASLifecycleAction -LifecycleHookName myLifecycleHook -
AutoScalingGroupName my-asg -LifecycleActionResult CONTINUE -LifecycleActionToken
 bcd2f1b8-9a78-44d3-8a7a-4dd07d7cf635
```

- Para obtener más información sobre la API, consulte CompleteLifecycleAction Herramientas de AWS para PowerShellCmdlet Reference (V4).

Herramientas para la versión 5 PowerShell

Ejemplo 1: En este ejemplo se completa la acción del ciclo de vida especificada.

```
Complete-ASLifecycleAction -LifecycleHookName myLifecycleHook -
AutoScalingGroupName my-asg -LifecycleActionResult CONTINUE -LifecycleActionToken
 bcd2f1b8-9a78-44d3-8a7a-4dd07d7cf635
```

- Para obtener más información sobre la API, consulte CompleteLifecycleActionla referencia de Herramientas de AWS para PowerShell cmdlets (V5).

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. Utilización de este servicio con un AWS SDK En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Úselo **`CreateAutoScalingGroup`** con un AWS SDK o CLI

Los siguientes ejemplos de código muestran cómo utilizar `CreateAutoScalingGroup`.

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:

- Conceptos básicos
- Creación y administración de un servicio resiliente

.NET

SDK para .NET (v4)

> **Note**
>
> Hay más información GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
/// <summary>
/// Create a new Amazon EC2 Auto Scaling group.
/// </summary>
/// <param name="groupName">The name to use for the new Auto Scaling
/// group.</param>
/// <param name="launchTemplateName">The name of the Amazon EC2 Auto Scaling
/// launch template to use to create instances in the group.</param>
/// <returns>A Boolean value indicating the success of the action.</returns>
public async Task<bool> CreateAutoScalingGroupAsync(
    string groupName,
    string launchTemplateName,
    string availabilityZone)
{
    var templateSpecification = new LaunchTemplateSpecification
    {
        LaunchTemplateName = launchTemplateName,
    };

    var zoneList = new List<string>
        {
            availabilityZone,
        };

    var request = new CreateAutoScalingGroupRequest
    {
        AutoScalingGroupName = groupName,
        AvailabilityZones = zoneList,
        LaunchTemplate = templateSpecification,
        MaxSize = 6,
        MinSize = 1
    };
```

```
        try
        {
            var response = await
_amazonAutoScaling.CreateAutoScalingGroupAsync(request);
            Console.WriteLine($"{groupName} Auto Scaling Group created");
            return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
        }
        catch (AlreadyExistsException)
        {
            Console.WriteLine($"{groupName} Auto Scaling Group already exists.");
            return true;
        }
    }
```

- Para obtener más información sobre la API, consulta CreateAutoScalingGroupla Referencia AWS SDK para .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::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region (overrides config file).
        // clientConfig.region = "us-east-1";

    Aws::AutoScaling::AutoScalingClient autoScalingClient(clientConfig);

        Aws::AutoScaling::Model::CreateAutoScalingGroupRequest request;
        request.SetAutoScalingGroupName(groupName);
        Aws::Vector<Aws::String> availabilityGroupZones;
        availabilityGroupZones.push_back(
                availabilityZones[availabilityZoneChoice - 1].GetZoneName());
        request.SetAvailabilityZones(availabilityGroupZones);
```

```
        request.SetMaxSize(1);
        request.SetMinSize(1);

        Aws::AutoScaling::Model::LaunchTemplateSpecification
launchTemplateSpecification;
        launchTemplateSpecification.SetLaunchTemplateName(templateName);
        request.SetLaunchTemplate(launchTemplateSpecification);

        Aws::AutoScaling::Model::CreateAutoScalingGroupOutcome outcome =
                autoScalingClient.CreateAutoScalingGroup(request);

        if (outcome.IsSuccess()) {
            std::cout << "Created Auto Scaling group '" << groupName << "'..."
                      << std::endl;
        }
        else if (outcome.GetError().GetErrorType() ==
                 Aws::AutoScaling::AutoScalingErrors::ALREADY_EXISTS_FAULT) {
            std::cout << "Auto Scaling group '" << groupName << "' already
exists."
                      << std::endl;
        }
        else {
            std::cerr << "Error with AutoScaling::CreateAutoScalingGroup. "
                      << outcome.GetError().GetMessage()
                      << std::endl;

        }
```

- Para obtener más información sobre la API, consulta CreateAutoScalingGroupla Referencia AWS SDK para C++ de la API.

CLI

AWS CLI

Ejemplo 1: crear un grupo de escalado automático

En el siguiente ejemplo `create-auto-scaling-group` se crea un grupo de escalado automático en subredes de varias zonas de disponibilidad dentro de una región. Las instancias se lanzan con la versión predeterminada de la plantilla de lanzamiento especificada. Tenga en cuenta que se utilizan valores predeterminados para la mayoría

de las demás configuraciones, como las políticas de terminación y la configuración de las comprobaciones de estado.

```
aws autoscaling create-auto-scaling-group \
    --auto-scaling-group-name my-asg \
    --launch-template LaunchTemplateId=lt-1234567890abcde12 \
    --min-size 1 \
    --max-size 5 \
    --vpc-zone-identifier "subnet-5ea0c127,subnet-6194ea3b,subnet-c934b782"
```

Este comando no genera ninguna salida.

Para obtener más información, consulte Grupos de escalado automático en la Guía del usuario de Amazon EC2 Auto Scaling.

Ejemplo 2: asociar un equilibrador de carga de aplicación, un equilibrador de carga de red o un equilibrador de carga de puerta de enlace

En este ejemplo, se especifica el ARN de un grupo de destino para un equilibrador de carga que admite el tráfico esperado. El tipo de comprobación de estado especifica `ELB` para que, cuando Elastic Load Balancing informa de una instancia como en mal estado, el grupo de escalado automático reemplaza la instancia. El comando también define un período de gracia de `600` segundos para la comprobación de estado. El período de gracia ayuda a evitar la finalización prematura de las instancias recién lanzadas.

```
aws autoscaling create-auto-scaling-group \
    --auto-scaling-group-name my-asg \
    --launch-template LaunchTemplateId=lt-1234567890abcde12 \
    --target-group-arns arn:aws:elasticloadbalancing:us-
west-2:123456789012:targetgroup/my-targets/943f017f100becff \
    --health-check-type ELB \
    --health-check-grace-period 600 \
    --min-size 1 \
    --max-size 5 \
    --vpc-zone-identifier "subnet-5ea0c127,subnet-6194ea3b,subnet-c934b782"
```

Este comando no genera ninguna salida.

Para obtener más información, consulte Elastic Load Balancing y Amazon EC2 Auto Scaling en la Guía del usuario de Amazon EC2 Auto Scaling.

Ejemplo 3: especificación de un grupo con ubicación y utilizar la versión más reciente de la plantilla de lanzamiento

En este ejemplo, se lanzan instancias a un grupo con ubicación en una única zona de disponibilidad. Esto puede resultar útil para grupos de baja latencia con cargas de trabajo de HPC. En este ejemplo, también se especifican el tamaño mínimo, el tamaño máximo y la capacidad deseada del grupo.

```
aws autoscaling create-auto-scaling-group \
    --auto-scaling-group-name my-asg \
    --launch-template LaunchTemplateId=lt-12345678890abcde12,Version='$Latest' \
    --min-size 1 \
    --max-size 5 \
    --desired-capacity 3 \
    --placement-group my-placement-group \
    --vpc-zone-identifier "subnet-6194ea3b"
```

Este comando no genera ninguna salida.

Para obtener más información, consulte Grupos de ubicación en la Guía del usuario de Amazon EC2 para instancias Linux.

Ejemplo 4: especificación de un grupo de escalado automático de una sola instancia y utilizar una versión específica de la plantilla de lanzamiento

En este ejemplo, se crea un grupo de escalado automático con una capacidad mínima y máxima establecida en 1 para garantizar que se ejecute una instancia. El comando también especifica la v1 de una plantilla de lanzamiento en la que se especifica el ID de un ENI existente. Cuando utilice una plantilla de lanzamiento que especifique un ENI existente para eth0, debe especificar una zona de disponibilidad para el grupo de escalado automático que coincida con la interfaz de red, sin especificar también un ID de subred en la solicitud.

```
aws autoscaling create-auto-scaling-group \
    --auto-scaling-group-name my-asg-single-instance \
    --launch-template LaunchTemplateName=my-template-for-auto-scaling,Version='1'
 \
    --min-size 1 \
    --max-size 1 \
    --availability-zones us-west-2a
```

Este comando no genera ninguna salida.

Para obtener más información, consulte Grupos de escalado automático en la Guía del usuario de Amazon EC2 Auto Scaling.

Ejemplo 5: especificación de una política de terminación diferente

En este ejemplo, se crea un grupo de escalado automático mediante una configuración de lanzamiento y se establece la política de terminación para terminar primero las instancias más antiguas. El comando también aplica una etiqueta al grupo y a sus instancias, con una clave de `Role` y un valor de `WebServer`.

```
aws autoscaling create-auto-scaling-group \
    --auto-scaling-group-name my-asg \
    --launch-configuration-name my-lc \
    --min-size 1 \
    --max-size 5 \
    --termination-policies "OldestInstance" \
    --tags "ResourceId=my-asg,ResourceType=auto-scaling-
group,Key=Role,Value=WebServer,PropagateAtLaunch=true" \
    --vpc-zone-identifier "subnet-5ea0c127,subnet-6194ea3b,subnet-c934b782"
```

Este comando no genera ninguna salida.

Para obtener más información, consulte Utilización de políticas de terminación de Amazon EC2 Auto Scaling en la Guía del usuario de Amazon EC2 Auto Scaling.

Ejemplo 6: especificación de un enlace de ciclo de vida de lanzamiento

En este ejemplo, se crea un grupo de escalado automático con un enlace de ciclo de vida que admite una acción personalizada cuando se lanza una instancia.

```
aws autoscaling create-auto-scaling-group \
    --cli-input-json file://~/config.json
```

Contenido del archivo `config.json`:

```
{
    "AutoScalingGroupName": "my-asg",
    "LaunchTemplate": {
        "LaunchTemplateId": "lt-1234567890abcde12"
    },
```

```
    "LifecycleHookSpecificationList": [{
        "LifecycleHookName": "my-launch-hook",
        "LifecycleTransition": "autoscaling:EC2_INSTANCE_LAUNCHING",
        "NotificationTargetARN": "arn:aws:sqs:us-west-2:123456789012:my-sqs-
queue",
        "RoleARN": "arn:aws:iam::123456789012:role/my-notification-role",
        "NotificationMetadata": "SQS message metadata",
        "HeartbeatTimeout": 4800,
        "DefaultResult": "ABANDON"
    }],
    "MinSize": 1,
    "MaxSize": 5,
    "VPCZoneIdentifier": "subnet-5ea0c127,subnet-6194ea3b,subnet-c934b782",
    "Tags": [{
        "ResourceType": "auto-scaling-group",
        "ResourceId": "my-asg",
        "PropagateAtLaunch": true,
        "Value": "test",
        "Key": "environment"
    }]
}
```

Este comando no genera ninguna salida.

Para obtener más información, consulte Enlaces de ciclo de vida de Amazon EC2 Auto Scaling en la guía del usuario de Amazon EC2 Auto Scaling.

Ejemplo 7: especificación de un enlace de ciclo de vida de terminación

En este ejemplo, se crea un grupo de escalado automático con un enlace de ciclo de vida que admite una acción personalizada en la terminación de una instancia.

```
aws autoscaling create-auto-scaling-group \
    --cli-input-json file://~/config.json
```

Contenido de `config.json`:

```
{
    "AutoScalingGroupName": "my-asg",
    "LaunchTemplate": {
        "LaunchTemplateId": "lt-1234567890abcde12"
    },
```

```
    "LifecycleHookSpecificationList": [{
        "LifecycleHookName": "my-termination-hook",
        "LifecycleTransition": "autoscaling:EC2_INSTANCE_TERMINATING",
        "HeartbeatTimeout": 120,
        "DefaultResult": "CONTINUE"
    }],
    "MinSize": 1,
    "MaxSize": 5,
    "TargetGroupARNs": [
        "arn:aws:elasticloadbalancing:us-west-2:123456789012:targetgroup/my-
targets/73e2d6bc24d8a067"
    ],
    "VPCZoneIdentifier": "subnet-5ea0c127,subnet-6194ea3b,subnet-c934b782"
}
```

Este comando no genera ninguna salida.

Para obtener más información, consulte Enlaces de ciclo de vida de Amazon EC2 Auto Scaling en la guía del usuario de Amazon EC2 Auto Scaling.

Ejemplo 8: especificación de una política de terminación personalizada

En este ejemplo, se crea un grupo de escalado automático que especifica una política de terminación de funciones de Lambda personalizada que indica a Amazon EC2 Auto Scaling qué instancias son seguras de terminar al escalarlas horizontalmente.

```
aws autoscaling create-auto-scaling-group \
    --auto-scaling-group-name my-asg-single-instance \
    --launch-template LaunchTemplateName=my-template-for-auto-scaling \
    --min-size 1 \
    --max-size 5 \
    --termination-policies "arn:aws:lambda:us-
west-2:123456789012:function:HelloFunction:prod" \
    --vpc-zone-identifier "subnet-5ea0c127,subnet-6194ea3b,subnet-c934b782"
```

Este comando no genera ninguna salida.

Para obtener más información, consulte Creación de una política de terminación personalizada con Lambda en la Guía del usuario de Amazon EC2 Auto Scaling.

- Para obtener más información sobre la API, consulta CreateAutoScalingGroupla 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.core.waiters.WaiterResponse;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.autoscaling.AutoScalingClient;
import software.amazon.awssdk.services.autoscaling.model.AutoScalingException;
import
 software.amazon.awssdk.services.autoscaling.model.CreateAutoScalingGroupRequest;
import
 software.amazon.awssdk.services.autoscaling.model.DescribeAutoScalingGroupsRequest;
import
 software.amazon.awssdk.services.autoscaling.model.DescribeAutoScalingGroupsResponse;
import
 software.amazon.awssdk.services.autoscaling.model.LaunchTemplateSpecification;
import software.amazon.awssdk.services.autoscaling.waiters.AutoScalingWaiter;

/**
 * Before running this SDK for Java (v2) code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class CreateAutoScalingGroup {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <groupName> <launchTemplateName> <serviceLinkedRoleARN>
 <vpcZoneId>

                Where:
```

```
                    groupName - The name of the Auto Scaling group.
                    launchTemplateName - The name of the launch template.\s
                    vpcZoneId - A subnet Id for a virtual private cloud (VPC)
where instances in the Auto Scaling group can be created.
                """;

        if (args.length != 3) {
            System.out.println(usage);
            System.exit(1);
        }

        String groupName = args[0];
        String launchTemplateName = args[1];
        String vpcZoneId = args[2];
        AutoScalingClient autoScalingClient = AutoScalingClient.builder()
                .region(Region.US_EAST_1)
                .build();

        createAutoScalingGroup(autoScalingClient, groupName, launchTemplateName,
vpcZoneId);
        autoScalingClient.close();
    }

    public static void createAutoScalingGroup(AutoScalingClient
autoScalingClient,
            String groupName,
            String launchTemplateName,
            String vpcZoneId) {

        try {
            AutoScalingWaiter waiter = autoScalingClient.waiter();
            LaunchTemplateSpecification templateSpecification =
LaunchTemplateSpecification.builder()
                    .launchTemplateName(launchTemplateName)
                    .build();

            CreateAutoScalingGroupRequest request =
CreateAutoScalingGroupRequest.builder()
                    .autoScalingGroupName(groupName)
                    .availabilityZones("us-east-1a")
                    .launchTemplate(templateSpecification)
                    .maxSize(1)
                    .minSize(1)
                    .vpcZoneIdentifier(vpcZoneId)
```

```
                    .build();

            autoScalingClient.createAutoScalingGroup(request);
            DescribeAutoScalingGroupsRequest groupsRequest =
 DescribeAutoScalingGroupsRequest.builder()
                    .autoScalingGroupNames(groupName)
                    .build();

            WaiterResponse<DescribeAutoScalingGroupsResponse> waiterResponse =
 waiter
                    .waitUntilGroupExists(groupsRequest);
            waiterResponse.matched().response().ifPresent(System.out::println);
            System.out.println("Auto Scaling Group created");

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

- Para obtener más información sobre la API, consulta CreateAutoScalingGroupla Referencia AWS SDK for Java 2.x 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 createAutoScalingGroup(
    groupName: String,
    launchTemplateNameVal: String,
    serviceLinkedRoleARNVal: String,
    vpcZoneIdVal: String,
) {
```

```
    val templateSpecification =
        LaunchTemplateSpecification {
            launchTemplateName = launchTemplateNameVal
        }

    val request =
        CreateAutoScalingGroupRequest {
            autoScalingGroupName = groupName
            availabilityZones = listOf("us-east-1a")
            launchTemplate = templateSpecification
            maxSize = 1
            minSize = 1
            vpcZoneIdentifier = vpcZoneIdVal
            serviceLinkedRoleArn = serviceLinkedRoleARNVal
        }

    // This object is required for the waiter call.
    val groupsRequestWaiter =
        DescribeAutoScalingGroupsRequest {
            autoScalingGroupNames = listOf(groupName)
        }

    AutoScalingClient { region = "us-east-1" }.use { autoScalingClient ->
        autoScalingClient.createAutoScalingGroup(request)
        autoScalingClient.waitUntilGroupExists(groupsRequestWaiter)
        println("$groupName was created!")
    }
}
```

- Para obtener más información sobre la API, consulta CreateAutoScalingGroupla referencia sobre el AWS SDK para la API de Kotlin.

PHP

SDK para PHP

> **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 function createAutoScalingGroup(
        $autoScalingGroupName,
        $availabilityZones,
        $minSize,
        $maxSize,
        $launchTemplateId
    ) {
        return $this->autoScalingClient->createAutoScalingGroup([
            'AutoScalingGroupName' => $autoScalingGroupName,
            'AvailabilityZones' => $availabilityZones,
            'MinSize' => $minSize,
            'MaxSize' => $maxSize,
            'LaunchTemplate' => [
                'LaunchTemplateId' => $launchTemplateId,
            ],
        ]);
    }
```

- Para obtener más información sobre la API, consulta CreateAutoScalingGroupla Referencia AWS SDK para PHP de la API.

PowerShell

Herramientas para la PowerShell versión 4

Ejemplo 1: En este ejemplo se crea un grupo de escalado automático con el nombre y los atributos especificados. La capacidad deseada por defecto es el tamaño mínimo. Por lo tanto, este grupo de escalado automático inicializa dos instancias, una en cada una de las dos zonas de disponibilidad especificadas.

```
New-ASAutoScalingGroup -AutoScalingGroupName my-asg -LaunchConfigurationName my-
lc -MinSize 2 -MaxSize 6 -AvailabilityZone @("us-west-2a", "us-west-2b")
```

- Para obtener más información sobre la API, consulte CreateAutoScalingGroup Herramientas de AWS para PowerShellCmdlet Reference (V4).

Herramientas para la versión 5 PowerShell

Ejemplo 1: En este ejemplo se crea un grupo de escalado automático con el nombre y los atributos especificados. La capacidad deseada por defecto es el tamaño mínimo. Por lo tanto,

este grupo de escalado automático inicializa dos instancias, una en cada una de las dos zonas de disponibilidad especificadas.

```
New-ASAutoScalingGroup -AutoScalingGroupName my-asg -LaunchConfigurationName my-
lc -MinSize 2 -MaxSize 6 -AvailabilityZone @("us-west-2a", "us-west-2b")
```

- Para obtener más información sobre la API, consulte CreateAutoScalingGroupla referencia de Herramientas de AWS para PowerShell cmdlets (V5).

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 AutoScalingWrapper:
    """Encapsulates Amazon EC2 Auto Scaling actions."""

    def __init__(self, autoscaling_client):
        """
        :param autoscaling_client: A Boto3 Amazon EC2 Auto Scaling client.
        """
        self.autoscaling_client = autoscaling_client


    def create_group(
        self,
        group_name: str,
        group_zones: List[str],
        launch_template_name: str,
        min_size: int,
        max_size: int,
    ) -> None:
        """
        Creates an Auto Scaling group.

        :param group_name: The name to give to the group.
```

```
        :param group_zones: The Availability Zones in which instances can be
created.
        :param launch_template_name: The name of an existing Amazon EC2 launch
template.
                                     The launch template specifies the
configuration of
                                     instances that are created by auto scaling
activities.
        :param min_size: The minimum number of active instances in the group.
        :param max_size: The maximum number of active instances in the group.
        :return: None
        :raises ClientError: If there is an error creating the Auto Scaling
group.
        """
        try:
            self.autoscaling_client.create_auto_scaling_group(
                AutoScalingGroupName=group_name,
                AvailabilityZones=group_zones,
                LaunchTemplate={
                    "LaunchTemplateName": launch_template_name,
                    "Version": "$Default",
                },
                MinSize=min_size,
                MaxSize=max_size,
            )

            # Wait for the group to exist.
            waiter = self.autoscaling_client.get_waiter("group_exists")
            waiter.wait(AutoScalingGroupNames=[group_name])

            logger.info(f"Successfully created Auto Scaling group {group_name}.")

        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            logger.error(f"Failed to create Auto Scaling group {group_name}.")
            if error_code == "AlreadyExistsFault":
                logger.error(
                    f"An Auto Scaling group with the name '{group_name}' already
exists. "
                    "Please use a different name or update the existing group.",
                )
            elif error_code == "LimitExceededFault":
                logger.error(
                    "The request failed because you have reached the limit "
```

```
                    "on the number of Auto Scaling groups or launch
 configurations. "
                    "Consider deleting unused resources or request a limit
 increase. "
                    "\nSee Auto Scaling Service Quota documentation here:"
                    "\n\thttps://docs.aws.amazon.com/autoscaling/ec2/userguide/
ec2-auto-scaling-quotas.html"
                )
            logger.error(f"Full error:\n\t{err}")
            raise
```

- Para obtener más información sobre la API, consulta CreateAutoScalingGroupla 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 create_group(client: &Client, name: &str, id: &str) -> Result<(), Error>
 {
    client
        .create_auto_scaling_group()
        .auto_scaling_group_name(name)
        .instance_id(id)
        .min_size(1)
        .max_size(5)
        .send()
        .await?;

    println!("Created AutoScaling group");

    Ok(())
}
```

- Para obtener más información sobre la API, consulta CreateAutoScalingGroupla referencia sobre la API de AWS SDK para Rust.

SAP ABAP

SDK para 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 lo_launch_template TYPE REF TO /aws1/cl_asclaunchtemplatespec.

    " Example: iv_group_name = 'my-auto-scaling-group'
    " Example: iv_launch_template_name = 'my-launch-template'
    " Example: iv_min_size = 1
    " Example: iv_max_size = 3
    " Example: iv_vpc_zone_identifier = 'subnet-12345,subnet-67890' (for VPC)

    TRY.
        " Create launch template specification
        lo_launch_template = NEW /aws1/cl_asclaunchtemplatespec(
          iv_launchtemplatename = iv_launch_template_name
          iv_version = '$Default' ).

        " Create the Auto Scaling group
        " Use VPCZoneIdentifier for VPC subnets, or AvailabilityZones for EC2-
Classic
        IF iv_vpc_zone_identifier IS NOT INITIAL.
          " VPC-based deployment - use subnet IDs
          ao_asc->createautoscalinggroup(
            iv_autoscalinggroupname = iv_group_name
            iv_vpczoneidentifier = iv_vpc_zone_identifier
            io_launchtemplate = lo_launch_template
            iv_minsize = iv_min_size
            iv_maxsize = iv_max_size ).
        ELSE.
```

```
          " EC2-Classic or default VPC - use availability zones
          ao_asc->createautoscalinggroup(
            iv_autoscalinggroupname = iv_group_name
            it_availabilityzones = it_group_zones
            io_launchtemplate = lo_launch_template
            iv_minsize = iv_min_size
            iv_maxsize = iv_max_size ).
        ENDIF.

        " Wait for the group to be created (simplified - in production use proper
polling)
        WAIT UP TO 10 SECONDS.

        MESSAGE 'Auto Scaling group created successfully' TYPE 'I'.

      CATCH /aws1/cx_ascalreadyexistsfault INTO DATA(lo_already_exists).
        RAISE EXCEPTION lo_already_exists.
      CATCH /aws1/cx_asclimitexceededfault INTO DATA(lo_limit_exceeded).
        RAISE EXCEPTION lo_limit_exceeded.
      CATCH /aws1/cx_rt_generic INTO DATA(lo_generic_exception).
        RAISE EXCEPTION lo_generic_exception.
    ENDTRY.
```

- Para obtener más información sobre la API, consulte CreateAutoScalingGroupla 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, consulteUtilización de este servicio con un AWS SDK. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Utilizar **`CreateLaunchConfiguration`** con una CLI

Los siguientes ejemplos de código muestran cómo utilizar `CreateLaunchConfiguration`.

CLI

AWS CLI

Ejemplo 1: Cómo crear una configuración de inicialización

En este ejemplo, se crea una configuración de inicialización sencilla.

```
aws autoscaling create-launch-configuration \
    --launch-configuration-name my-lc \
    --image-id ami-04d5cc9b88example \
    --instance-type m5.large
```

Este comando no genera ninguna salida.

Para obtener más información, consulte Crear una configuración de lanzamiento en la Guía del usuario de Amazon EC2 Auto Scaling.

Ejemplo 2: Para crear una configuración de inicio con un grupo de seguridad, un key pair y un script de arranque

En este ejemplo, se crea una configuración de inicio con un grupo de seguridad, un key pair y un script de arranque incluidos en los datos del usuario.

```
aws autoscaling create-launch-configuration \
    --launch-configuration-name my-lc \
    --image-id ami-04d5cc9b88example \
    --instance-type m5.large \
    --security-groups sg-eb2af88example \
    --key-name my-key-pair \
    --user-data file://myuserdata.txt
```

Este comando no genera ninguna salida.

Para obtener más información, consulte Crear una configuración de lanzamiento en la Guía del usuario de Amazon EC2 Auto Scaling.

Ejemplo 3: crear una configuración de inicialización con un rol de IAM

En este ejemplo, se crea una configuración de inicialización con el nombre del perfil de instancia de un rol de IAM.

```
aws autoscaling create-launch-configuration \
    --launch-configuration-name my-lc \
    --image-id ami-04d5cc9b88example \
    --instance-type m5.large \
    --iam-instance-profile my-autoscaling-role
```

Este comando no genera ninguna salida.

Para obtener más información, consulte el tema Rol de IAM para las aplicaciones que se ejecutan en instancias Amazon EC2 en la guía del usuario de Amazon EC2 Auto Scaling.

Ejemplo 4: cómo habilitar la configuración de inicialización con la supervisión detallada habilitada

En este ejemplo, se crea una configuración de lanzamiento con la supervisión detallada de EC2 habilitada, que envía las métricas de EC2 en períodos de 1 minuto. CloudWatch

```
aws autoscaling create-launch-configuration \
    --launch-configuration-name my-lc \
    --image-id ami-04d5cc9b88example \
    --instance-type m5.large \
    --instance-monitoring Enabled=true
```

Este comando no genera ninguna salida.

Para obtener más información, consulte Configuración del monitoreo para instancias de Auto Scaling en la Guía del usuario de Amazon EC2 Auto Scaling.

Ejemplo 5: cómo crear una configuración de inicialización que inicialice instancias de spot

En este ejemplo, se crea una configuración de inicialización que utiliza instancias de spot como única opción de compra.

```
aws autoscaling create-launch-configuration \
    --launch-configuration-name my-lc \
    --image-id ami-04d5cc9b88example \
    --instance-type m5.large \
    --spot-price "0.50"
```

Este comando no genera ninguna salida.

Para obtener más información, consulte Solicitud de instancias de spot en la Guía del usuario de Amazon EC2 Auto Scaling.

Ejemplo 6: cómo crear una configuración de inicialización con una instancia de EC2

En este ejemplo, se crea una configuración de inicialización basada en los atributos de una instancia existente. Anula la tenencia de la ubicación y si se establece una dirección IP pública al incluir las opciones `--placement-tenancy` y`--no-associate-public-ip-address`.

```
aws autoscaling create-launch-configuration \
    --launch-configuration-name my-lc-from-instance \
    --instance-id i-0123a456700123456 \
    --instance-type m5.large \
    --no-associate-public-ip-address \
    --placement-tenancy dedicated
```

Este comando no genera ninguna salida.

Para obtener más información, consulte Crear una configuración de lanzamiento con una instancia EC2 en la Guía del usuario de Amazon EC2 Auto Scaling.

Ejemplo 7: cómo crear una configuración de inicialización con una asignación de dispositivos de bloques para un volumen de Amazon EBS

En este ejemplo, se crea una configuración de inicialización con una asignación de dispositivos de bloques para un volumen gp3 de Amazon EBS con el nombre del dispositivo `/dev/sdh` y un tamaño de volumen de 20.

```
aws autoscaling create-launch-configuration \
    --launch-configuration-name my-lc \
    --image-id ami-04d5cc9b88example \
    --instance-type m5.large \
    --block-device-mappings '[{"DeviceName":"/dev/sdh","Ebs":
{"VolumeSize":20,"VolumeType":"gp3"}}]'
```

Este comando no genera ninguna salida.

Para obtener más información, consulte EBS en la Referencia de la API de Amazon EC2 Auto Scaling.

Para obtener información sobre la sintaxis para citar valores de parámetros con formato JSON, consulte Uso de comillas con cadenas en la AWS CLI en la Guía del usuario de la interfaz de línea de AWS comandos.

Ejemplo 8: cómo crear una configuración de inicialización con una asignación de dispositivos de bloques para un volumen de almacén de instancias

En este ejemplo, se crea una configuración de inicialización con `ephemeral1` como un volumen de almacén de instancias con el nombre de dispositivo `/dev/sdc`.

```
aws autoscaling create-launch-configuration \
    --launch-configuration-name my-lc \
    --image-id ami-04d5cc9b88example \
    --instance-type m5.large \
    --block-device-mappings '[{"DeviceName":"/dev/
sdc","VirtualName":"ephemeral1"}]'
```

Este comando no genera ninguna salida.

Para obtener más información, consulte la BlockDeviceMappingreferencia de la API Amazon EC2 Auto Scaling.

Para obtener información sobre la sintaxis para citar valores de parámetros con formato JSON, consulte Uso de comillas con cadenas en la AWS CLI en la Guía del usuario de la interfaz de línea de AWS comandos.

Ejemplo 9: cómo crear una configuración de inicialización y suprimir un dispositivo de bloques se conecte en el momento de la inicialización

En este ejemplo, se crea una configuración de inicialización que suprime un dispositivo de bloques especificado por la asignación de dispositivos de bloques de la AMI (por ejemplo,`/dev/sdf`).

```
aws autoscaling create-launch-configuration \
    --launch-configuration-name my-lc \
    --image-id ami-04d5cc9b88example \
    --instance-type m5.large \
    --block-device-mappings '[{"DeviceName":"/dev/sdf","NoDevice":""}]'
```

Este comando no genera ninguna salida.

Para obtener más información, consulte la BlockDeviceMappingreferencia de la API Amazon EC2 Auto Scaling.

Para obtener información sobre la sintaxis para citar valores de parámetros con formato JSON, consulte Uso de comillas con cadenas en la AWS CLI en la Guía del usuario de la interfaz de línea de AWS comandos.

- Para obtener más información sobre la API, consulte CreateLaunchConfigurationla Referencia de comandos.AWS CLI

PowerShell

Herramientas para la PowerShell versión 4

Ejemplo 1: En este ejemplo se crea una configuración de inicialización denominada «my-lc». Las instancias de EC2 inicializadas por los grupos de escalado automático que utilizan esta configuración de inicialización utilizan un tipo de instancia, una AMI, un grupo de seguridad y un rol de IAM específicos.

```
New-ASLaunchConfiguration -LaunchConfigurationName my-lc -InstanceType
 "m3.medium" -ImageId "ami-12345678" -SecurityGroup "sg-12345678" -
IamInstanceProfile "myIamRole"
```

- Para obtener más información sobre la API, consulte CreateLaunchConfiguration Herramientas de AWS para PowerShellCmdlet Reference (V4).

Herramientas para la versión 5 PowerShell

Ejemplo 1: En este ejemplo se crea una configuración de inicialización denominada «my-lc». Las instancias de EC2 inicializadas por los grupos de escalado automático que utilizan esta configuración de inicialización utilizan un tipo de instancia, una AMI, un grupo de seguridad y un rol de IAM específicos.

```
New-ASLaunchConfiguration -LaunchConfigurationName my-lc -InstanceType
 "m3.medium" -ImageId "ami-12345678" -SecurityGroup "sg-12345678" -
IamInstanceProfile "myIamRole"
```

- Para obtener más información sobre la API, consulte CreateLaunchConfigurationla referencia de Herramientas de AWS para PowerShell cmdlets (V5).

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. Utilización de este servicio con un AWS SDK En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Utilizar **`CreateOrUpdateTags`** con una CLI

Los siguientes ejemplos de código muestran cómo utilizar `CreateOrUpdateTags`.

CLI

AWS CLI

Cómo crear o modificar etiquetas en un grupo de escalado automático

En este ejemplo, se agregan dos etiquetas al grupo de escalado automático especificado.

```
aws autoscaling create-or-update-tags \
    --tags ResourceId=my-asg,ResourceType=auto-scaling-
group,Key=Role,Value=WebServer,PropagateAtLaunch=true ResourceId=my-
asg,ResourceType=auto-scaling-
group,Key=Dept,Value=Research,PropagateAtLaunch=true
```

Este comando no genera ninguna salida.

Para obtener más información, consulte Etiquetado de grupos e instancias de Auto Scaling en la Guía del usuario de Amazon EC2 Auto Scaling.

- Para obtener más información sobre la API, consulte CreateOrUpdateTagsla Referencia de AWS CLI comandos.

PowerShell

Herramientas para la PowerShell versión 4

Ejemplo 1: En este ejemplo se agrega una etiqueta única al grupo de escalado automático especificado. La clave de la etiqueta es 'myTag' y el valor de la etiqueta es 'myTagValue'. El escalado automático propaga esta etiqueta a las siguientes instancias de EC2 inicializadas por el grupo de escalado automático. La sintaxis utilizada en este ejemplo requiere la PowerShell versión 3 o posterior.

```
Set-ASTag -Tag @( @{ResourceType="auto-scaling-group"; ResourceId="my-asg";
 Key="myTag"; Value="myTagValue"; PropagateAtLaunch=$true} )
```

Ejemplo 2: Con la PowerShell versión 2, debe usar New-Object para crear la etiqueta para el parámetro Tag.

```
$tag = New-Object Amazon.AutoScaling.Model.Tag
$tag.ResourceType = "auto-scaling-group"
$tag.ResourceId = "my-asg"
```

```
$tag.Key = "myTag"
$tag.Value = "myTagValue"
$tag.PropagateAtLaunch = $true
Set-ASTag -Tag $tag
```

- Para obtener más información sobre la API, consulte CreateOrUpdateTags Herramientas de AWS para PowerShellCmdlet Reference (V4).

Herramientas para la versión 5 PowerShell

Ejemplo 1: En este ejemplo se agrega una etiqueta única al grupo de escalado automático especificado. La clave de la etiqueta es 'myTag' y el valor de la etiqueta es ". myTagValue El escalado automático propaga esta etiqueta a las siguientes instancias de EC2 inicializadas por el grupo de escalado automático. La sintaxis utilizada en este ejemplo requiere la PowerShell versión 3 o posterior.

```
Set-ASTag -Tag @( @{ResourceType="auto-scaling-group"; ResourceId="my-asg";
 Key="myTag"; Value="myTagValue"; PropagateAtLaunch=$true} )
```

Ejemplo 2: Con la PowerShell versión 2, debe usar New-Object para crear la etiqueta para el parámetro Tag.

```
$tag = New-Object Amazon.AutoScaling.Model.Tag
$tag.ResourceType = "auto-scaling-group"
$tag.ResourceId = "my-asg"
$tag.Key = "myTag"
$tag.Value = "myTagValue"
$tag.PropagateAtLaunch = $true
Set-ASTag -Tag $tag
```

- Para obtener más información sobre la API, consulte CreateOrUpdateTags Herramientas de AWS para PowerShellCmdlet Reference (V5).

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. Utilización de este servicio con un AWS SDK En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Úselo **`DeleteAutoScalingGroup`** con un AWS SDK o CLI

Los siguientes ejemplos de código muestran cómo utilizar `DeleteAutoScalingGroup`.

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:

- Conceptos básicos
- Creación y administración de un servicio resiliente

.NET

SDK para .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.

Actualice el tamaño mínimo de un grupo de escalado automático a cero, finalice todas las instancias del grupo y elimine el grupo.

```
    /// <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>
    /// Delete an Auto Scaling group.
    /// </summary>
    /// <param name="groupName">The name of the Amazon EC2 Auto Scaling group.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteAutoScalingGroupAsync(
```

```
        string groupName)
    {
        var deleteAutoScalingGroupRequest = new DeleteAutoScalingGroupRequest
        {
            AutoScalingGroupName = groupName,
            ForceDelete = true,
        };

        var response = await
_amazonAutoScaling.DeleteAutoScalingGroupAsync(deleteAutoScalingGroupRequest);
        if (response.HttpStatusCode == System.Net.HttpStatusCode.OK)
        {
            Console.WriteLine($"You successfully deleted {groupName}");
            return true;
        }

        Console.WriteLine($"Couldn't delete {groupName}.");
        return false;
    }
```

- Para obtener más información sobre la API, consulta DeleteAutoScalingGroupla Referencia AWS SDK para .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::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region (overrides config file).
        // clientConfig.region = "us-east-1";

    Aws::AutoScaling::AutoScalingClient autoScalingClient(clientConfig);
```

```
            Aws::AutoScaling::Model::DeleteAutoScalingGroupRequest request;
            request.SetAutoScalingGroupName(groupName);

            Aws::AutoScaling::Model::DeleteAutoScalingGroupOutcome outcome =
                    autoScalingClient.DeleteAutoScalingGroup(request);

            if (outcome.IsSuccess()) {
                std::cout << "Auto Scaling group '" << groupName << "' was
deleted."
                          << std::endl;
            }
            else {
                std::cerr << "Error with AutoScaling::DeleteAutoScalingGroup. "
                          << outcome.GetError().GetMessage()
                          << std::endl;
                result = false;
            }
        }
```

- Para obtener más información sobre la API, consulta DeleteAutoScalingGroupla Referencia AWS SDK para C++ de la API.

CLI

AWS CLI

Ejemplo 1: eliminar el grupo de escalado automático especificado

En este ejemplo, se elimina el grupo de escalado automático especificado.

```
aws autoscaling delete-auto-scaling-group \
    --auto-scaling-group-name my-asg
```

Este comando no genera ninguna salida.

Para obtener más información, consulte Eliminación de la infraestructura de Auto Scaling en la Guía del usuario de Amazon EC2 Auto Scaling.

Ejemplo 2: para forzar la eliminación del grupo de escalado automático especificado

Para eliminar el grupo de escalado automático sin esperar a que las instancias del grupo terminen, utilice la opción `--force-delete`.

```
aws autoscaling delete-auto-scaling-group \
    --auto-scaling-group-name my-asg \
    --force-delete
```

Este comando no genera ninguna salida.

Para obtener más información, consulte Eliminación de la infraestructura de Auto Scaling en la Guía del usuario de Amazon EC2 Auto Scaling.

- Para obtener más información sobre la API, consulta DeleteAutoScalingGroupla 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.autoscaling.AutoScalingClient;
import software.amazon.awssdk.services.autoscaling.model.AutoScalingException;
import
 software.amazon.awssdk.services.autoscaling.model.DeleteAutoScalingGroupRequest;

/**
 * Before running this SDK for Java (v2) code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class DeleteAutoScalingGroup {
```

```
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <groupName>

                Where:
                    groupName - The name of the Auto Scaling group.
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String groupName = args[0];
        AutoScalingClient autoScalingClient = AutoScalingClient.builder()
                .region(Region.US_EAST_1)
                .build();

        deleteAutoScalingGroup(autoScalingClient, groupName);
        autoScalingClient.close();
    }

    public static void deleteAutoScalingGroup(AutoScalingClient
 autoScalingClient, String groupName) {
        try {
            DeleteAutoScalingGroupRequest deleteAutoScalingGroupRequest =
 DeleteAutoScalingGroupRequest.builder()
                    .autoScalingGroupName(groupName)
                    .forceDelete(true)
                    .build();


 autoScalingClient.deleteAutoScalingGroup(deleteAutoScalingGroupRequest);
            System.out.println("You successfully deleted " + groupName);

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

- Para obtener más información sobre la API, consulta DeleteAutoScalingGroupla Referencia AWS SDK for Java 2.x 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 deleteSpecificAutoScalingGroup(groupName: String) {
    val deleteAutoScalingGroupRequest =
        DeleteAutoScalingGroupRequest {
            autoScalingGroupName = groupName
            forceDelete = true
        }

    AutoScalingClient { region = "us-east-1" }.use { autoScalingClient ->
        autoScalingClient.deleteAutoScalingGroup(deleteAutoScalingGroupRequest)
        println("You successfully deleted $groupName")
    }
}
```

- Para obtener más información sobre la API, consulta DeleteAutoScalingGroupla referencia sobre el AWS SDK para la API de Kotlin.

PHP

SDK para PHP

> **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 function deleteAutoScalingGroup($autoScalingGroupName)
    {
        return $this->autoScalingClient->deleteAutoScalingGroup([
            'AutoScalingGroupName' => $autoScalingGroupName,
            'ForceDelete' => true,
        ]);
    }
```

- Para obtener más información sobre la API, consulta DeleteAutoScalingGroupla Referencia AWS SDK para PHP de la API.

PowerShell

Herramientas para la PowerShell versión 4

Ejemplo 1: En este ejemplo se elimina el grupo de escalado automático especificado si no tiene instancias en ejecución. Se le solicitará una confirmación antes de continuar con la operación.

```
Remove-ASAutoScalingGroup -AutoScalingGroupName my-asg
```

Salida:

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-ASAutoScalingGroup (DeleteAutoScalingGroup)" on
 Target "my-asg".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
 "Y"):
```

Ejemplo 2: Si especifica el parámetro Force, no se le solicitará una confirmación antes de continuar con la operación.

```
Remove-ASAutoScalingGroup -AutoScalingGroupName my-asg -Force
```

Ejemplo 3: En este ejemplo se elimina el grupo de escalado automático especificado y se finalizan todas las instancias en ejecución que contenga.

```
Remove-ASAutoScalingGroup -AutoScalingGroupName my-asg -ForceDelete $true -Force
```

- Para obtener más información sobre la API, consulte DeleteAutoScalingGroup Herramientas de AWS para PowerShellCmdlet Reference (V4).

Herramientas para la versión 5 PowerShell

Ejemplo 1: En este ejemplo se elimina el grupo de escalado automático especificado si no tiene instancias en ejecución. Se le solicitará una confirmación antes de continuar con la operación.

```
Remove-ASAutoScalingGroup -AutoScalingGroupName my-asg
```

Salida:

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-ASAutoScalingGroup (DeleteAutoScalingGroup)" on
 Target "my-asg".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
 "Y"):
```

Ejemplo 2: Si especifica el parámetro Force, no se le solicitará una confirmación antes de continuar con la operación.

```
Remove-ASAutoScalingGroup -AutoScalingGroupName my-asg -Force
```

Ejemplo 3: En este ejemplo se elimina el grupo de escalado automático especificado y se finalizan todas las instancias en ejecución que contenga.

```
Remove-ASAutoScalingGroup -AutoScalingGroupName my-asg -ForceDelete $true -Force
```

- Para obtener más información sobre la API, consulte DeleteAutoScalingGroupla referencia de Herramientas de AWS para PowerShell cmdlets (V5).

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.

Actualice el tamaño mínimo de un grupo de escalado automático a cero, finalice todas las instancias del grupo y elimine el grupo.

```
class AutoScalingWrapper:
    """
    Encapsulates Amazon EC2 Auto Scaling and EC2 management actions.
    """

    def __init__(
        self,
        resource_prefix: str,
        inst_type: str,
        ami_param: str,
        autoscaling_client: boto3.client,
        ec2_client: boto3.client,
        ssm_client: boto3.client,
        iam_client: boto3.client,
    ):
        """
        Initializes the AutoScaler class with the necessary parameters.

        :param resource_prefix: The prefix for naming AWS resources that are
 created by this class.
        :param inst_type: The type of EC2 instance to create, such as t3.micro.
        :param ami_param: The Systems Manager parameter used to look up the AMI
 that is created.
        :param autoscaling_client: A Boto3 EC2 Auto Scaling client.
        :param ec2_client: A Boto3 EC2 client.
```

```
        :param ssm_client: A Boto3 Systems Manager client.
        :param iam_client: A Boto3 IAM client.
        """
        self.inst_type = inst_type
        self.ami_param = ami_param
        self.autoscaling_client = autoscaling_client
        self.ec2_client = ec2_client
        self.ssm_client = ssm_client
        self.iam_client = iam_client
        sts_client = boto3.client("sts")
        self.account_id = sts_client.get_caller_identity()["Account"]

        self.key_pair_name = f"{resource_prefix}-key-pair"
        self.launch_template_name = f"{resource_prefix}-template-"
        self.group_name = f"{resource_prefix}-group"

        # Happy path
        self.instance_policy_name = f"{resource_prefix}-pol"
        self.instance_role_name = f"{resource_prefix}-role"
        self.instance_profile_name = f"{resource_prefix}-prof"

        # Failure mode
        self.bad_creds_policy_name = f"{resource_prefix}-bc-pol"
        self.bad_creds_role_name = f"{resource_prefix}-bc-role"
        self.bad_creds_profile_name = f"{resource_prefix}-bc-prof"


    def delete_autoscaling_group(self, group_name: str) -> None:
        """
        Terminates all instances in the group, then deletes the EC2 Auto Scaling
group.

        :param group_name: The name of the group to delete.
        """
        try:
            response = self.autoscaling_client.describe_auto_scaling_groups(
                AutoScalingGroupNames=[group_name]
            )
            groups = response.get("AutoScalingGroups", [])
            if len(groups) > 0:
                self.autoscaling_client.update_auto_scaling_group(
                    AutoScalingGroupName=group_name, MinSize=0
                )
```

```
                instance_ids = [inst["InstanceId"] for inst in groups[0]
["Instances"]]
                for inst_id in instance_ids:
                    self.terminate_instance(inst_id)

                # Wait for all instances to be terminated
                if instance_ids:
                    waiter = self.ec2_client.get_waiter("instance_terminated")
                    log.info("Waiting for all instances to be terminated...")
                    waiter.wait(InstanceIds=instance_ids)
                    log.info("All instances have been terminated.")
            else:
                log.info(f"No groups found named '{group_name}'! Nothing to do.")
        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            log.error(f"Failed to delete Auto Scaling group '{group_name}'.")
            if error_code == "ScalingActivityInProgressFault":
                log.error(
                    "Scaling activity is currently in progress. "
                    "Wait for the scaling activity to complete before attempting
 to delete the group again."
                )
            elif error_code == "ResourceContentionFault":
                log.error(
                    "The request failed due to a resource contention issue. "
                    "Ensure that no conflicting operations are being performed on
 the group."
                )
            log.error(f"Full error:\n\t{err}")
```

- Para obtener más información sobre la API, consulta DeleteAutoScalingGroupla 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 delete_group(client: &Client, name: &str, force: bool) -> Result<(),
 Error> {
    client
        .delete_auto_scaling_group()
        .auto_scaling_group_name(name)
        .set_force_delete(if force { Some(true) } else { None })
        .send()
        .await?;

    println!("Deleted Auto Scaling group");

    Ok(())
}
```

- Para obtener más información sobre la API, consulta DeleteAutoScalingGroupla referencia sobre la API de AWS SDK para Rust.

SAP ABAP

SDK para 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.

```
    " Example: iv_group_name = 'my-auto-scaling-group'
```

```
    TRY.
        ao_asc->deleteautoscalinggroup(
          iv_autoscalinggroupname = iv_group_name ).

        " Wait for the group to be deleted (simplified - in production use proper
polling)
        WAIT UP TO 10 SECONDS.

        MESSAGE 'Auto Scaling group deleted successfully' TYPE 'I'.

      CATCH /aws1/cx_ascscaactivityinprg00 INTO DATA(lo_activity_in_progress).
        RAISE EXCEPTION lo_activity_in_progress.
      CATCH /aws1/cx_ascresourceinusefault INTO DATA(lo_resource_in_use).
        RAISE EXCEPTION lo_resource_in_use.
      CATCH /aws1/cx_rt_generic INTO DATA(lo_generic_exception).
        RAISE EXCEPTION lo_generic_exception.
    ENDTRY.
```

- Para obtener más información sobre la API, consulte DeleteAutoScalingGroupla 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, consulteUtilización de este servicio con un AWS SDK. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Utilizar **`DeleteLaunchConfiguration`** con una CLI

Los siguientes ejemplos de código muestran cómo utilizar `DeleteLaunchConfiguration`.

CLI

AWS CLI

Cómo eliminar una configuración de inicialización

En este ejemplo se elimina la configuración de inicialización especificada.

```
aws autoscaling delete-launch-configuration \
    --launch-configuration-name my-launch-config
```

Este comando no genera ninguna salida.

Para obtener más información, consulte Eliminación de la infraestructura de Auto Scaling en la Guía del usuario de Amazon EC2 Auto Scaling.

- Para obtener más información sobre la API, consulte DeleteLaunchConfigurationla Referencia de AWS CLI comandos.

PowerShell

Herramientas para la PowerShell versión 4

Ejemplo 1: En este ejemplo se elimina la configuración de inicialización especificada si no está asociada a un grupo de escalado automático. Se le solicitará una confirmación antes de continuar con la operación.

```
Remove-ASLaunchConfiguration -LaunchConfigurationName my-lc
```

Salida:

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-ASLaunchConfiguration (DeleteLaunchConfiguration)"
 on Target "my-lc".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
 "Y"):
```

Ejemplo 2: Si especifica el parámetro Force, no se le solicitará una confirmación antes de continuar con la operación.

```
Remove-ASLaunchConfiguration -LaunchConfigurationName my-lc -Force
```

- Para obtener más información sobre la API, consulte DeleteLaunchConfiguration Herramientas de AWS para PowerShellCmdlet Reference (V4).

Herramientas para la versión 5 PowerShell

Ejemplo 1: En este ejemplo se elimina la configuración de inicialización especificada si no está asociada a un grupo de escalado automático. Se le solicitará una confirmación antes de continuar con la operación.

```
Remove-ASLaunchConfiguration -LaunchConfigurationName my-lc
```

Salida:

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-ASLaunchConfiguration (DeleteLaunchConfiguration)"
 on Target "my-lc".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
 "Y"):
```

Ejemplo 2: Si especifica el parámetro Force, no se le solicitará una confirmación antes de continuar con la operación.

```
Remove-ASLaunchConfiguration -LaunchConfigurationName my-lc -Force
```

- Para obtener más información sobre la API, consulte DeleteLaunchConfigurationla referencia de Herramientas de AWS para PowerShell cmdlets (V5).

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. Utilización de este servicio con un AWS SDK En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Utilizar **`DeleteLifecycleHook`** con una CLI

Los siguientes ejemplos de código muestran cómo utilizar `DeleteLifecycleHook`.

CLI

AWS CLI

Cómo eliminar un enlace de ciclo de vida

En este ejemplo se elimina el enlace de ciclo de vida especificado.

```
aws autoscaling delete-lifecycle-hook \
    --lifecycle-hook-name my-lifecycle-hook \
    --auto-scaling-group-name my-asg
```

Este comando no genera ninguna salida.

- Para obtener más información sobre la API, consulte DeleteLifecycleHookla Referencia de AWS CLI comandos.

PowerShell

Herramientas para la PowerShell versión 4

Ejemplo 1: En este ejemplo se elimina el enlace de ciclo de vida especificado para el grupo de escalado automático especificado. Se le solicitará una confirmación antes de continuar con la operación.

```
Remove-ASLifecycleHook -AutoScalingGroupName my-asg -LifecycleHookName
 myLifecycleHook
```

Salida:

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-ASLifecycleHook (DeleteLifecycleHook)" on Target
 "myLifecycleHook".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
 "Y"):
```

Ejemplo 2: Si especifica el parámetro Force, no se le solicitará una confirmación antes de continuar con la operación.

```
Remove-ASLifecycleHook -AutoScalingGroupName my-asg -LifecycleHookName
 myLifecycleHook -Force
```

- Para obtener más información sobre la API, consulte DeleteLifecycleHook Herramientas de AWS para PowerShellCmdlet Reference (V4).

Herramientas para la versión 5 PowerShell

Ejemplo 1: En este ejemplo se elimina el enlace de ciclo de vida especificado para el grupo de escalado automático especificado. Se le solicitará una confirmación antes de continuar con la operación.

```
Remove-ASLifecycleHook -AutoScalingGroupName my-asg -LifecycleHookName
 myLifecycleHook
```

Salida:

```
Confirm
```

```
Are you sure you want to perform this action?
Performing operation "Remove-ASLifecycleHook (DeleteLifecycleHook)" on Target
 "myLifecycleHook".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
 "Y"):
```

Ejemplo 2: Si especifica el parámetro Force, no se le solicitará una confirmación antes de continuar con la operación.

```
Remove-ASLifecycleHook -AutoScalingGroupName my-asg -LifecycleHookName
 myLifecycleHook -Force
```

- Para obtener más información sobre la API, consulte DeleteLifecycleHookla referencia de Herramientas de AWS para PowerShell cmdlets (V5).

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. Utilización de este servicio con un AWS SDK En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Utilizar **`DeleteNotificationConfiguration`** con una CLI

Los siguientes ejemplos de código muestran cómo utilizar `DeleteNotificationConfiguration`.

CLI

AWS CLI

Cómo eliminar una notificación de escalado automático

En este ejemplo, se elimina la notificación especificada del grupo de escalado automático especificado.

```
aws autoscaling delete-notification-configuration \
    --auto-scaling-group-name my-asg \
    --topic-arn arn:aws:sns:us-west-2:123456789012:my-sns-topic
```

Este comando no genera ninguna salida.

Para obtener más información, consulte Delete the notification configuration en la Guía del usuario de Amazon EC2 Auto Scaling.

- Para obtener más información sobre la API, consulte DeleteNotificationConfigurationla Referencia de AWS CLI comandos.

PowerShell

Herramientas para la PowerShell versión 4

Ejemplo 1: En este ejemplo se elimina la acción de notificación especificada. Se le solicitará una confirmación antes de continuar con la operación.

```
Remove-ASNotificationConfiguration -AutoScalingGroupName my-asg -TopicARN
 "arn:aws:sns:us-west-2:123456789012:my-topic"
```

Salida:

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-ASNotificationConfiguration
 (DeleteNotificationConfiguration)" on Target
"arn:aws:sns:us-west-2:123456789012:my-topic".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
 "Y"):
```

Ejemplo 2: Si especifica el parámetro Force, no se le solicitará una confirmación antes de continuar con la operación.

```
Remove-ASNotificationConfiguration -AutoScalingGroupName my-asg -TopicARN
 "arn:aws:sns:us-west-2:123456789012:my-topic" -Force
```

- Para obtener más información sobre la API, consulte DeleteNotificationConfiguration Herramientas de AWS para PowerShellCmdlet Reference (V4).

Herramientas para la versión 5 PowerShell

Ejemplo 1: En este ejemplo se elimina la acción de notificación especificada. Se le solicitará una confirmación antes de continuar con la operación.

```
Remove-ASNotificationConfiguration -AutoScalingGroupName my-asg -TopicARN
 "arn:aws:sns:us-west-2:123456789012:my-topic"
```

Salida:

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-ASNotificationConfiguration
 (DeleteNotificationConfiguration)" on Target
"arn:aws:sns:us-west-2:123456789012:my-topic".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
 "Y"):
```

Ejemplo 2: Si especifica el parámetro Force, no se le solicitará una confirmación antes de continuar con la operación.

```
Remove-ASNotificationConfiguration -AutoScalingGroupName my-asg -TopicARN
 "arn:aws:sns:us-west-2:123456789012:my-topic" -Force
```

- Para obtener más información sobre la API, consulte DeleteNotificationConfigurationla referencia de Herramientas de AWS para PowerShell cmdlets (V5).

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. Utilización de este servicio con un AWS SDK En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Utilizar **`DeletePolicy`** con una CLI

Los siguientes ejemplos de código muestran cómo utilizar `DeletePolicy`.

CLI

AWS CLI

Cómo eliminar una política de escalado

En este ejemplo, se elimina el grupo de escalado automático especificado.

```
aws autoscaling delete-policy \
    --auto-scaling-group-name my-asg \
    --policy-name alb1000-target-tracking-scaling-policy
```

Este comando no genera ninguna salida.

- Para obtener más información sobre la API, consulte DeletePolicyla Referencia de AWS CLI comandos.

PowerShell

Herramientas para la PowerShell versión 4

Ejemplo 1: En este ejemplo se elimina la política especificada para el grupo de escalado automático especificado. Se le solicitará una confirmación antes de continuar con la operación.

```
Remove-ASPolicy -AutoScalingGroupName my-asg -PolicyName myScaleInPolicy
```

Salida:

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-ASPolicy (DeletePolicy)" on Target
 "myScaleInPolicy".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
 "Y"):
```

Ejemplo 2: Si especifica el parámetro Force, no se le solicitará una confirmación antes de continuar con la operación.

```
Remove-ASPolicy -AutoScalingGroupName my-asg -PolicyName myScaleInPolicy -Force
```

- Para obtener más información sobre la API, consulte DeletePolicy Herramientas de AWS para PowerShellCmdlet Reference (V4).

Herramientas para la versión 5 PowerShell

Ejemplo 1: En este ejemplo se elimina la política especificada para el grupo de escalado automático especificado. Se le solicitará una confirmación antes de continuar con la operación.

```
Remove-ASPolicy -AutoScalingGroupName my-asg -PolicyName myScaleInPolicy
```

Salida:

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-ASPolicy (DeletePolicy)" on Target
 "myScaleInPolicy".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
 "Y"):
```

Ejemplo 2: Si especifica el parámetro Force, no se le solicitará una confirmación antes de continuar con la operación.

```
Remove-ASPolicy -AutoScalingGroupName my-asg -PolicyName myScaleInPolicy -Force
```

- Para obtener más información sobre la API, consulte DeletePolicyla referencia de Herramientas de AWS para PowerShell cmdlets (V5).

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. Utilización de este servicio con un AWS SDK En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Utilizar **`DeleteScheduledAction`** con una CLI

Los siguientes ejemplos de código muestran cómo utilizar `DeleteScheduledAction`.

CLI

AWS CLI

Cómo crear una acción programada para un grupo de escalado automático

En este ejemplo, se elimina la acción programada especificada del grupo de escalado automático especificado.

```
aws autoscaling delete-scheduled-action \
    --auto-scaling-group-name my-asg \
    --scheduled-action-name my-scheduled-action
```

Este comando no genera ninguna salida.

- Para obtener más información sobre la API, consulte DeleteScheduledActionla Referencia de AWS CLI comandos.

PowerShell

Herramientas para la PowerShell versión 4

Ejemplo 1: En este ejemplo se elimina la acción programada especificada para el grupo de escalado automático especificado. Se le solicitará una confirmación antes de continuar con la operación.

```
Remove-ASScheduledAction -AutoScalingGroupName my-asg -ScheduledAction
 "myScheduledAction"
```

Salida:

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-ASScheduledAction (DeleteScheduledAction)" on Target
 "myScheduledAction".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
 "Y"):
```

Ejemplo 2: Si especifica el parámetro Force, no se le solicitará una confirmación antes de continuar con la operación.

```
Remove-ASScheduledAction -AutoScalingGroupName my-asg -ScheduledAction
 "myScheduledAction" -Force
```

- Para obtener más información sobre la API, consulte DeleteScheduledAction Herramientas de AWS para PowerShellCmdlet Reference (V4).

Herramientas para la versión 5 PowerShell

Ejemplo 1: En este ejemplo se elimina la acción programada especificada para el grupo de escalado automático especificado. Se le solicitará una confirmación antes de continuar con la operación.

```
Remove-ASScheduledAction -AutoScalingGroupName my-asg -ScheduledAction
 "myScheduledAction"
```

Salida:

```
Confirm
```

```
Are you sure you want to perform this action?
Performing operation "Remove-ASScheduledAction (DeleteScheduledAction)" on Target
 "myScheduledAction".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
 "Y"):
```

Ejemplo 2: Si especifica el parámetro Force, no se le solicitará una confirmación antes de continuar con la operación.

```
Remove-ASScheduledAction -AutoScalingGroupName my-asg -ScheduledAction
 "myScheduledAction" -Force
```

- Para obtener más información sobre la API, consulte DeleteScheduledActionla referencia de Herramientas de AWS para PowerShell cmdlets (V5).

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. Utilización de este servicio con un AWS SDK En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Utilizar **`DeleteTags`** con una CLI

Los siguientes ejemplos de código muestran cómo utilizar `DeleteTags`.

CLI

AWS CLI

Cómo eliminar una etiqueta de un grupo de escalado automático

En este ejemplo, se elimina la etiqueta especificada del grupo de escalado automático especificado.

```
aws autoscaling delete-tags \
    --tags ResourceId=my-asg,ResourceType=auto-scaling-
group,Key=Dept,Value=Research
```

Este comando no genera ninguna salida.

Para obtener más información, consulte Etiquetado de grupos e instancias de Auto Scaling en la Guía del usuario de Amazon EC2 Auto Scaling.

- Para obtener más información sobre la API, consulte DeleteTagsla Referencia de AWS CLI comandos.

PowerShell

Herramientas para la PowerShell versión 4

Ejemplo 1: En este ejemplo se elimina la etiqueta especificada del grupo de escalado automático especificado. Se le solicitará una confirmación antes de continuar con la operación. La sintaxis utilizada en este ejemplo requiere PowerShell la versión 3 o posterior.

```
Remove-ASTag -Tag @( @{ResourceType="auto-scaling-group"; ResourceId="my-asg";
 Key="myTag" } )
```

Salida:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-ASTag (DeleteTags)" on target
 "Amazon.AutoScaling.Model.Tag".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
 "Y"):
```

Ejemplo 2: Si especifica el parámetro Force, no se le solicitará una confirmación antes de continuar con la operación.

```
Remove-ASTag -Tag @( @{ResourceType="auto-scaling-group"; ResourceId="my-asg";
 Key="myTag" } ) -Force
```

Ejemplo 3: Con la versión 2 de PowerShell, debe usar New-Object para crear la etiqueta para el parámetro Etiqueta.

```
$tag = New-Object Amazon.AutoScaling.Model.Tag
$tag.ResourceType = "auto-scaling-group"
$tag.ResourceId = "my-asg"
$tag.Key = "myTag"
Remove-ASTag -Tag $tag -Force
```

- Para obtener información sobre la API, consulte DeleteTags Herramientas de AWS para PowerShellCmdlet Reference (V4).

Herramientas para la versión 5 PowerShell

Ejemplo 1: En este ejemplo se elimina la etiqueta especificada del grupo de escalado automático especificado. Se le solicitará una confirmación antes de continuar con la operación. La sintaxis utilizada en este ejemplo requiere la PowerShell versión 3 o posterior.

```
Remove-ASTag -Tag @( @{ResourceType="auto-scaling-group"; ResourceId="my-asg";
 Key="myTag" } )
```

Salida:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-ASTag (DeleteTags)" on target
 "Amazon.AutoScaling.Model.Tag".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
 "Y"):
```

Ejemplo 2: Si especifica el parámetro Force, no se le solicitará una confirmación antes de continuar con la operación.

```
Remove-ASTag -Tag @( @{ResourceType="auto-scaling-group"; ResourceId="my-asg";
 Key="myTag" } ) -Force
```

Ejemplo 3: Con la versión 2 de PowerShell, debe usar New-Object para crear la etiqueta para el parámetro Etiqueta.

```
$tag = New-Object Amazon.AutoScaling.Model.Tag
$tag.ResourceType = "auto-scaling-group"
$tag.ResourceId = "my-asg"
$tag.Key = "myTag"
Remove-ASTag -Tag $tag -Force
```

- Para obtener más información sobre la API, consulte DeleteTags Herramientas de AWS para PowerShellCmdlet Reference (V5).

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. Utilización de este servicio con un AWS SDK En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Utilizar **`DescribeAccountLimits`** con una CLI

Los siguientes ejemplos de código muestran cómo utilizar `DescribeAccountLimits`.

CLI

AWS CLI

Cómo describir los límites de cuenta de Amazon EC2 Auto Scaling

En este ejemplo, se describen los límites de Amazon EC2 Auto Scaling para AWS su cuenta.

```
aws autoscaling describe-account-limits
```

Salida:

```
{
    "NumberOfLaunchConfigurations": 5,
    "MaxNumberOfLaunchConfigurations": 100,
    "NumberOfAutoScalingGroups": 3,
    "MaxNumberOfAutoScalingGroups": 20
}
```

Para obtener más información, consulte Amazon EC2 Auto Scaling service quotas en la Guía del usuario de Amazon EC2 Auto Scaling.

- Para obtener más información sobre la API, consulte DescribeAccountLimitsla Referencia de AWS CLI comandos.

PowerShell

Herramientas para la PowerShell versión 4

Ejemplo 1: Este ejemplo describe los límites de recursos de Auto Scaling para su AWS cuenta.

```
Get-ASAccountLimit
```

Salida:

```
MaxNumberOfAutoScalingGroups    : 20
```

```
MaxNumberOfLaunchConfigurations : 100
```

- Para obtener más información sobre la API, consulte DescribeAccountLimits Herramientas de AWS para PowerShellCmdlet Reference (V4).

Herramientas para la versión 5 PowerShell

Ejemplo 1: Este ejemplo describe los límites de recursos de Auto Scaling para su AWS cuenta.

```
Get-ASAccountLimit
```

Salida:

```
MaxNumberOfAutoScalingGroups    : 20
MaxNumberOfLaunchConfigurations : 100
```

- Para obtener más información sobre la API, consulte DescribeAccountLimits Herramientas de AWS para PowerShellCmdlet Reference (V5).

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. Utilización de este servicio con un AWS SDK En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Utilizar **`DescribeAdjustmentTypes`** con una CLI

Los siguientes ejemplos de código muestran cómo utilizar `DescribeAdjustmentTypes`.

CLI

AWS CLI

Cómo describir los tipos de ajuste de escalado disponibles

En este ejemplo se describen los tipos de ajuste disponibles.

```
aws autoscaling describe-adjustment-types
```

Salida:

```
{
    "AdjustmentTypes": [
```

```
        {
            "AdjustmentType": "ChangeInCapacity"
        },
        {
            "AdjustmentType": "ExactCapacity"
        },
        {
            "AdjustmentType": "PercentChangeInCapacity"
        }
    ]
}
```

Para obtener más información, consulte la sección sobre tipos de ajustes de de escalado en la guía del usuario de Amazon EC2 Auto Scaling.

- Para obtener más información sobre la API, consulte DescribeAdjustmentTypesla Referencia de AWS CLI comandos.

PowerShell

Herramientas para la PowerShell versión 4

Ejemplo 1: En este ejemplo se describen los tipos de ajuste que son compatibles con el escalado automático.

```
Get-ASAdjustmentType
```

Salida:

```
Type
----
ChangeInCapacity
ExactCapacity
PercentChangeInCapacity
```

- Para obtener más información sobre la API, consulte DescribeAdjustmentTypes Herramientas de AWS para PowerShellCmdlet Reference (V4).

Herramientas para la versión 5 PowerShell

Ejemplo 1: En este ejemplo se describen los tipos de ajuste que son compatibles con el escalado automático.

```
Get-ASAdjustmentType
```

Salida:

```
Type
----
ChangeInCapacity
ExactCapacity
PercentChangeInCapacity
```

- Para obtener más información sobre la API, consulte DescribeAdjustmentTypesla referencia de Herramientas de AWS para PowerShell cmdlets (V5).

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. Utilización de este servicio con un AWS SDK En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Úselo **`DescribeAutoScalingGroups`** con un AWS SDK o CLI

Los siguientes ejemplos de código muestran cómo utilizar `DescribeAutoScalingGroups`.

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:

- Conceptos básicos
- Creación y administración de un servicio resiliente

.NET

SDK para .NET (v4)

> **Note**
>
> Hay más información GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

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

```
    /// Get data about the instances in an Amazon EC2 Auto Scaling group.
    /// </summary>
    /// <param name="groupName">The name of the Amazon EC2 Auto Scaling group.</
param>
    /// <returns>A list of Amazon EC2 Auto Scaling details.</returns>
    public async Task<List<AutoScalingInstanceDetails>>
 DescribeAutoScalingInstancesAsync(
        string groupName)
    {
        var groups = await DescribeAutoScalingGroupsAsync(groupName);
        var instanceIds = new List<string>();
        var instanceDetails = new List<AutoScalingInstanceDetails>();
        if (groups != null)
        {
            groups.ForEach(group =>
            {
                if (group.AutoScalingGroupName == groupName && group.Instances !=
 null)
                {
                    group.Instances.ForEach(instance =>
                    {
                        instanceIds.Add(instance.InstanceId);
                    });
                }
            });

            var scalingGroupsRequest = new DescribeAutoScalingInstancesRequest
            {
                MaxRecords = 10,
                InstanceIds = instanceIds,
            };

            var response =
                await _amazonAutoScaling.DescribeAutoScalingInstancesAsync(
                    scalingGroupsRequest);
            if (response.AutoScalingInstances != null)
            {
                instanceDetails = response.AutoScalingInstances;
            }
        }

        return instanceDetails;
    }
```

- Para obtener más información sobre la API, consulta DescribeAutoScalingGroupsla Referencia AWS SDK para .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::Client::ClientConfiguration clientConfig;
    // Optional: Set to the AWS Region (overrides config file).
    // clientConfig.region = "us-east-1";

Aws::AutoScaling::AutoScalingClient autoScalingClient(clientConfig);

Aws::AutoScaling::Model::DescribeAutoScalingGroupsRequest request;
Aws::Vector<Aws::String> groupNames;
groupNames.push_back(groupName);
request.SetAutoScalingGroupNames(groupNames);

Aws::AutoScaling::Model::DescribeAutoScalingGroupsOutcome outcome =
        client.DescribeAutoScalingGroups(request);

if (outcome.IsSuccess()) {
    autoScalingGroup = outcome.GetResult().GetAutoScalingGroups();
}
else {
    std::cerr << "Error with AutoScaling::DescribeAutoScalingGroups. "
              << outcome.GetError().GetMessage()
              << std::endl;
}
```

- Para obtener más información sobre la API, consulta DescribeAutoScalingGroupsla Referencia AWS SDK para C++ de la API.

CLI

AWS CLI

Ejemplo 1: Descripción del grupo de escalado automático especificado

En este ejemplo, se describe el grupo de escalado automático especificado.

```
aws autoscaling describe-auto-scaling-groups \
    --auto-scaling-group-names my-asg
```

Salida:

```
{
    "AutoScalingGroups": [
        {
            "AutoScalingGroupName": "my-asg",
            "AutoScalingGroupARN": "arn:aws:autoscaling:us-
west-2:123456789012:autoScalingGroup:930d940e-891e-4781-
a11a-7b0acd480f03:autoScalingGroupName/my-asg",
            "LaunchTemplate": {
                "LaunchTemplateName": "my-launch-template",
                "Version": "1",
                "LaunchTemplateId": "lt-1234567890abcde12"
            },
            "MinSize": 0,
            "MaxSize": 1,
            "DesiredCapacity": 1,
            "DefaultCooldown": 300,
            "AvailabilityZones": [
                "us-west-2a",
                "us-west-2b",
                "us-west-2c"
            ],
            "LoadBalancerNames": [],
            "TargetGroupARNs": [],
            "HealthCheckType": "EC2",
            "HealthCheckGracePeriod": 0,
            "Instances": [
                {
                    "InstanceId": "i-06905f55584de02da",
                    "InstanceType": "t2.micro",
                    "AvailabilityZone": "us-west-2a",
```

```
                    "HealthStatus": "Healthy",
                    "LifecycleState": "InService",
                    "ProtectedFromScaleIn": false,
                    "LaunchTemplate": {
                        "LaunchTemplateName": "my-launch-template",
                        "Version": "1",
                        "LaunchTemplateId": "lt-1234567890abcde12"
                    }
                }
            ],
            "CreatedTime": "2023-10-28T02:39:22.152Z",
            "SuspendedProcesses": [],
            "VPCZoneIdentifier": "subnet-5ea0c127,subnet-6194ea3b,subnet-
c934b782",
            "EnabledMetrics": [],
            "Tags": [],
            "TerminationPolicies": [
                "Default"
            ],
            "NewInstancesProtectedFromScaleIn": false,
            "ServiceLinkedRoleARN":"arn",
            "TrafficSources": []
        }
    ]
}
```

Ejemplo 2: Descripción de los 100 primeros grupos de escalado automático especificados

En este ejemplo, se describen los grupos de escalado automático especificados. Le permite especificar hasta 100 nombres de grupos.

```
aws autoscaling describe-auto-scaling-groups \
    --max-items 100 \
    --auto-scaling-group-names "group1" "group2" "group3" "group4"
```

Consulte el ejemplo 1 para ver una salida de muestra.

Ejemplo 3: descripción de un grupo de escalado automático en la región especificada

En este ejemplo, se describen los grupos de escalado automático en la región especificada, hasta un máximo de 75 grupos.

```
aws autoscaling describe-auto-scaling-groups \
```

```
    --max-items 75 \
    --region us-east-1
```

Consulte el ejemplo 1 para ver una salida de muestra.

Ejemplo 4: descripción del número especificado de grupos de escalado automático

Para devolver un número específico de grupos de escalado automático, utilice la opción `--max-items`.

```
aws autoscaling describe-auto-scaling-groups \
    --max-items 1
```

Consulte el ejemplo 1 para ver una salida de muestra.

Si la salida incluye un campo `NextToken`, hay más grupos. Para obtener los grupos adicionales, utilice el valor de este campo con la opción `--starting-token` en una llamada posterior de la siguiente manera.

```
aws autoscaling describe-auto-scaling-groups \
    --starting-token Z3M3LMPEXAMPLE
```

Consulte el ejemplo 1 para ver una salida de muestra.

Ejemplo 5: cómo describir grupos de escalado automático que utilizan configuraciones de inicialización

En este ejemplo, se utiliza la opción `--query` para describir los grupos de escalado automático que utilizan configuraciones de inicialización.

```
aws autoscaling describe-auto-scaling-groups \
    --query 'AutoScalingGroups[?LaunchConfigurationName!=`null`]'
```

Salida:

```
[
    {
        "AutoScalingGroupName": "my-asg",
        "AutoScalingGroupARN": "arn:aws:autoscaling:us-
west-2:123456789012:autoScalingGroup:930d940e-891e-4781-
a11a-7b0acd480f03:autoScalingGroupName/my-asg",
```

```
        "LaunchConfigurationName": "my-lc",
        "MinSize": 0,
        "MaxSize": 1,
        "DesiredCapacity": 1,
        "DefaultCooldown": 300,
        "AvailabilityZones": [
            "us-west-2a",
            "us-west-2b",
            "us-west-2c"
        ],
        "LoadBalancerNames": [],
        "TargetGroupARNs": [],
        "HealthCheckType": "EC2",
        "HealthCheckGracePeriod": 0,
        "Instances": [
            {
                "InstanceId": "i-088c57934a6449037",
                "InstanceType": "t2.micro",
                "AvailabilityZone": "us-west-2c",
                "HealthStatus": "Healthy",
                "LifecycleState": "InService",
                "LaunchConfigurationName": "my-lc",
                "ProtectedFromScaleIn": false
            }
        ],
        "CreatedTime": "2023-10-28T02:39:22.152Z",
        "SuspendedProcesses": [],
        "VPCZoneIdentifier": "subnet-5ea0c127,subnet-6194ea3b,subnet-c934b782",
        "EnabledMetrics": [],
        "Tags": [],
        "TerminationPolicies": [
            "Default"
        ],
        "NewInstancesProtectedFromScaleIn": false,
        "ServiceLinkedRoleARN":"arn",
        "TrafficSources": []
    }
]
```


Para obtener más información, consulte Filtrar la salida AWS CLI en la Guía del usuario de la interfaz de línea de AWS comandos.

- Para obtener más información sobre la API, consulte DescribeAutoScalingGroupsla 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.autoscaling.AutoScalingClient;
import software.amazon.awssdk.services.autoscaling.model.AutoScalingException;
import software.amazon.awssdk.services.autoscaling.model.AutoScalingGroup;
import
 software.amazon.awssdk.services.autoscaling.model.DescribeAutoScalingGroupsResponse;
import
 software.amazon.awssdk.services.autoscaling.model.DescribeAutoScalingGroupsRequest;
import software.amazon.awssdk.services.autoscaling.model.Instance;
import java.util.List;

/**
 * Before running this SDK for Java (v2) code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class DescribeAutoScalingInstances {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <groupName>

                Where:
                    groupName - The name of the Auto Scaling group.
                """;

        if (args.length != 1) {
```

```
            System.out.println(usage);
            System.exit(1);
        }

        String groupName = args[0];
        AutoScalingClient autoScalingClient = AutoScalingClient.builder()
                .region(Region.US_EAST_1)
                .build();

        String instanceId = getAutoScaling(autoScalingClient, groupName);
        System.out.println(instanceId);
        autoScalingClient.close();
    }

    public static String getAutoScaling(AutoScalingClient autoScalingClient,
String groupName) {
        try {
            String instanceId = "";
            DescribeAutoScalingGroupsRequest scalingGroupsRequest =
DescribeAutoScalingGroupsRequest.builder()
                    .autoScalingGroupNames(groupName)
                    .build();

            DescribeAutoScalingGroupsResponse response = autoScalingClient
                    .describeAutoScalingGroups(scalingGroupsRequest);
            List<AutoScalingGroup> groups = response.autoScalingGroups();
            for (AutoScalingGroup group : groups) {
                System.out.println("The group name is " +
group.autoScalingGroupName());
                System.out.println("The group ARN is " +
group.autoScalingGroupARN());

                List<Instance> instances = group.instances();
                for (Instance instance : instances) {
                    instanceId = instance.instanceId();
                }
            }
            return instanceId;
        } catch (AutoScalingException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return "";
    }
```

```
}
```

- Para obtener más información sobre la API, consulta DescribeAutoScalingGroupsla Referencia AWS SDK for Java 2.x 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 getAutoScalingGroups(groupName: String) {
    val scalingGroupsRequest =
        DescribeAutoScalingGroupsRequest {
            autoScalingGroupNames = listOf(groupName)
        }

    AutoScalingClient { region = "us-east-1" }.use { autoScalingClient ->
        val response =
 autoScalingClient.describeAutoScalingGroups(scalingGroupsRequest)
        response.autoScalingGroups?.forEach { group ->
            println("The group name is ${group.autoScalingGroupName}")
            println("The group ARN is ${group.autoScalingGroupArn}")
            group.instances?.forEach { instance ->
                println("The instance id is ${instance.instanceId}")
                println("The lifecycle state is " + instance.lifecycleState)
            }
        }
    }
}
```

- Para obtener más información sobre la API, consulta DescribeAutoScalingGroupsla referencia sobre el AWS SDK para la API de Kotlin.

PHP

SDK para PHP

> **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 function describeAutoScalingGroups($autoScalingGroupNames)
    {
        return $this->autoScalingClient->describeAutoScalingGroups([
            'AutoScalingGroupNames' => $autoScalingGroupNames
        ]);
    }
```

- Para obtener más información sobre la API, consulta DescribeAutoScalingGroupsla Referencia AWS SDK para PHP de la API.

PowerShell

Herramientas para la PowerShell versión 4

Ejemplo 1: En este ejemplo se enumeran los nombres de los grupos de escalado automático.

```
Get-ASAutoScalingGroup | format-table -property AutoScalingGroupName
```

Salida:

```
AutoScalingGroupName
--------------------
my-asg-1
my-asg-2
my-asg-3
my-asg-4
my-asg-5
my-asg-6
```

Ejemplo 2: En este ejemplo, se describe el grupo de escalado automático especificado.

```
Get-ASAutoScalingGroup -AutoScalingGroupName my-asg-1
```

Salida:

```
AutoScalingGroupARN     : arn:aws:autoscaling:us-
west-2:123456789012:autoScalingGroup:930d940e-891e-4781-a11a-7b0acd480
                          f03:autoScalingGroupName/my-asg-1
AutoScalingGroupName    : my-asg-1
AvailabilityZones       : {us-west-2b, us-west-2a}
CreatedTime             : 3/1/2015 9:05:31 AM
DefaultCooldown         : 300
DesiredCapacity         : 2
EnabledMetrics          : {}
HealthCheckGracePeriod  : 300
HealthCheckType         : EC2
Instances               : {my-lc}
LaunchConfigurationName : my-lc
LoadBalancerNames       : {}
MaxSize                 : 0
MinSize                 : 0
PlacementGroup          :
Status                  :
SuspendedProcesses      : {}
Tags                    : {}
TerminationPolicies     : {Default}
VPCZoneIdentifier       : subnet-e4f33493,subnet-5264e837
```

Ejemplo 3: En este ejemplo, se describen los dos grupos de escalado automático especificados.

```
Get-ASAutoScalingGroup -AutoScalingGroupName @("my-asg-1", "my-asg-2")
```

Ejemplo 4: En este ejemplo, se describen las instancias de escalado automático para el grupo de escalado automático especificado.

```
(Get-ASAutoScalingGroup -AutoScalingGroupName my-asg-1).Instances
```

Ejemplo 5: En este ejemplo se describen todos los grupos de escalado automático.

```
Get-ASAutoScalingGroup
```

Ejemplo 6: Este LaunchTemplate ejemplo describe el grupo de Auto Scaling especificado. En este ejemplo, se supone que las «Opciones de compra de instancias» están configuradas en «Adherirse a la plantilla de inicialización». En caso de que esta opción esté configurada como «Combinar opciones de compra y tipos de instancias», se LaunchTemplate puede acceder a ella mediante»MixedInstancesPolicy. LaunchTemplate«propiedad.

```
(Get-ASAutoScalingGroup -AutoScalingGroupName my-ag-1).LaunchTemplate
```

Salida:

```
LaunchTemplateId     LaunchTemplateName   Version
----------------     ------------------   -------
lt-06095fd619cb40371 test-launch-template $Default
```

- Para obtener más información sobre la API, consulte DescribeAutoScalingGroups Herramientas de AWS para PowerShellCmdlet Reference (V4).

Herramientas para la versión 5 PowerShell

Ejemplo 1: En este ejemplo se enumeran los nombres de los grupos de escalado automático.

```
Get-ASAutoScalingGroup | format-table -property AutoScalingGroupName
```

Salida:

```
AutoScalingGroupName
--------------------
my-asg-1
my-asg-2
my-asg-3
my-asg-4
my-asg-5
my-asg-6
```

Ejemplo 2: En este ejemplo, se describe el grupo de escalado automático especificado.

```
Get-ASAutoScalingGroup -AutoScalingGroupName my-asg-1
```

Salida:

```
AutoScalingGroupARN     : arn:aws:autoscaling:us-
west-2:123456789012:autoScalingGroup:930d940e-891e-4781-a11a-7b0acd480
                          f03:autoScalingGroupName/my-asg-1
AutoScalingGroupName    : my-asg-1
AvailabilityZones       : {us-west-2b, us-west-2a}
CreatedTime             : 3/1/2015 9:05:31 AM
DefaultCooldown         : 300
DesiredCapacity         : 2
EnabledMetrics          : {}
HealthCheckGracePeriod  : 300
HealthCheckType         : EC2
Instances               : {my-lc}
LaunchConfigurationName : my-lc
LoadBalancerNames       : {}
MaxSize                 : 0
MinSize                 : 0
PlacementGroup          :
Status                  :
SuspendedProcesses      : {}
Tags                    : {}
TerminationPolicies     : {Default}
VPCZoneIdentifier       : subnet-e4f33493,subnet-5264e837
```

Ejemplo 3: En este ejemplo, se describen los dos grupos de escalado automático especificados.

```
Get-ASAutoScalingGroup -AutoScalingGroupName @("my-asg-1", "my-asg-2")
```

Ejemplo 4: En este ejemplo, se describen las instancias de escalado automático para el grupo de escalado automático especificado.

```
(Get-ASAutoScalingGroup -AutoScalingGroupName my-asg-1).Instances
```

Ejemplo 5: En este ejemplo se describen todos los grupos de escalado automático.

```
Get-ASAutoScalingGroup
```

Ejemplo 6: Este LaunchTemplate ejemplo describe el grupo de Auto Scaling especificado. En este ejemplo, se supone que las «Opciones de compra de instancias» están configuradas en

«Adherirse a la plantilla de inicialización». En caso de que esta opción esté configurada como «Combinar opciones de compra y tipos de instancias», se LaunchTemplate puede acceder a ella mediante»MixedInstancesPolicy. LaunchTemplate«propiedad.

```
(Get-ASAutoScalingGroup -AutoScalingGroupName my-ag-1).LaunchTemplate
```

Salida:

```
LaunchTemplateId     LaunchTemplateName   Version
----------------     ------------------   -------
lt-06095fd619cb40371 test-launch-template $Default
```

- Para obtener más información sobre la API, consulte DescribeAutoScalingGroups Herramientas de AWS para PowerShellCmdlet Reference (V5).

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 AutoScalingWrapper:
    """Encapsulates Amazon EC2 Auto Scaling actions."""

    def __init__(self, autoscaling_client):
        """
        :param autoscaling_client: A Boto3 Amazon EC2 Auto Scaling client.
        """
        self.autoscaling_client = autoscaling_client


    def describe_group(self, group_name: str) -> Optional[Dict[str, Any]]:
        """
        Gets information about an Auto Scaling group.
```

```
        :param group_name: The name of the group to look up.
        :return: A dictionary with information about the group if found,
otherwise None.
        :raises ClientError: If there is an error describing the Auto Scaling
group.
        """
        try:
            paginator = self.autoscaling_client.get_paginator(
                "describe_auto_scaling_groups"
            )
            response_iterator =
paginator.paginate(AutoScalingGroupNames=[group_name])
            groups = []
            for response in response_iterator:
                groups.extend(response.get("AutoScalingGroups", []))

            logger.info(
                f"Successfully retrieved information for Auto Scaling group
{group_name}."
            )

        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            logger.error(f"Failed to describe Auto Scaling group {group_name}.")
            if error_code == "ResourceContentionFault":
                logger.error(
                    "There is a conflict with another operation that is modifying
the "
                    f"Auto Scaling group '{group_name}' Please try again later."
                )
            logger.error(f"Full error:\n\t{err}")
            raise
        else:
            return groups[0] if len(groups) > 0 else None
```

- Para obtener más información sobre la API, consulta DescribeAutoScalingGroupsla 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 list_groups(client: &Client) -> Result<(), Error> {
    let resp = client.describe_auto_scaling_groups().send().await?;

    println!("Groups:");

    let groups = resp.auto_scaling_groups();

    for group in groups {
        println!(
            "Name:  {}",
            group.auto_scaling_group_name().unwrap_or("Unknown")
        );
        println!(
            "Arn:   {}",
            group.auto_scaling_group_arn().unwrap_or("unknown"),
        );
        println!("Zones: {:?}", group.availability_zones(),);
        println!();
    }

    println!("Found {} group(s)", groups.len());

    Ok(())
}
```

- Para obtener más información sobre la API, consulta DescribeAutoScalingGroupsla referencia sobre la API de AWS SDK para Rust.

SAP ABAP

SDK para 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_group_names TYPE /aws1/
cl_ascautoscgroupnames_w=>tt_autoscalinggroupnames.
    DATA lo_group_name TYPE REF TO /aws1/cl_ascautoscgroupnames_w.

    " Example: iv_group_name = 'my-auto-scaling-group'

    TRY.
        " Build group names parameter
        CREATE OBJECT lo_group_name
          EXPORTING
            iv_value = iv_group_name.
        APPEND lo_group_name TO lt_group_names.

        " Describe the Auto Scaling group
        DATA(lo_output) = ao_asc->describeautoscalinggroups(
          it_autoscalinggroupnames = lt_group_names ).

        " Return the first (and only) group in the result
        DATA(lt_groups) = lo_output->get_autoscalinggroups( ).
        IF lines( lt_groups ) > 0.
          READ TABLE lt_groups INDEX 1 INTO DATA(lo_group).
          oo_output = lo_group.
        ENDIF.

        MESSAGE 'Auto Scaling group information retrieved successfully' TYPE 'I'.

      CATCH /aws1/cx_ascresrccontionfault INTO DATA(lo_contention).
        RAISE EXCEPTION lo_contention.
      CATCH /aws1/cx_rt_generic INTO DATA(lo_generic_exception).
        RAISE EXCEPTION lo_generic_exception.
    ENDTRY.
```

- Para obtener más información sobre la API, consulte DescribeAutoScalingGroupsla 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, consulteUtilización de este servicio con un AWS SDK. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Úselo **`DescribeAutoScalingInstances`** con un AWS SDK o CLI

Los siguientes ejemplos de código muestran cómo utilizar `DescribeAutoScalingInstances`.

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:

- Conceptos básicos

.NET

SDK para .NET (v4)

> **Note**
>
> Hay más información GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
    /// <summary>
    /// Get data about the instances in an Amazon EC2 Auto Scaling group.
    /// </summary>
    /// <param name="groupName">The name of the Amazon EC2 Auto Scaling group.</
param>
    /// <returns>A list of Amazon EC2 Auto Scaling details.</returns>
    public async Task<List<AutoScalingInstanceDetails>>
 DescribeAutoScalingInstancesAsync(
        string groupName)
    {
        var groups = await DescribeAutoScalingGroupsAsync(groupName);
        var instanceIds = new List<string>();
        var instanceDetails = new List<AutoScalingInstanceDetails>();
```

```
        if (groups != null)
        {
            groups.ForEach(group =>
            {
                if (group.AutoScalingGroupName == groupName && group.Instances != 
null)
                {
                    group.Instances.ForEach(instance =>
                    {
                        instanceIds.Add(instance.InstanceId);
                    });
                }
            });

            var scalingGroupsRequest = new DescribeAutoScalingInstancesRequest
            {
                MaxRecords = 10,
                InstanceIds = instanceIds,
            };

            var response =
                await _amazonAutoScaling.DescribeAutoScalingInstancesAsync(
                    scalingGroupsRequest);
            if (response.AutoScalingInstances != null)
            {
                instanceDetails = response.AutoScalingInstances;
            }
        }

        return instanceDetails;
    }
```

- Para obtener más información sobre la API, consulta DescribeAutoScalingInstancesla Referencia AWS SDK para .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::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region (overrides config file).
        // clientConfig.region = "us-east-1";

    Aws::AutoScaling::AutoScalingClient autoScalingClient(clientConfig);

        Aws::AutoScaling::Model::DescribeAutoScalingInstancesRequest request;
        request.SetInstanceIds(instanceIDs);

        Aws::AutoScaling::Model::DescribeAutoScalingInstancesOutcome outcome =
                client.DescribeAutoScalingInstances(request);

        if (outcome.IsSuccess()) {
            const
Aws::Vector<Aws::AutoScaling::Model::AutoScalingInstanceDetails>
&instancesDetails =
                    outcome.GetResult().GetAutoScalingInstances();

        }
        else {
            std::cerr << "Error with AutoScaling::DescribeAutoScalingInstances. "
                      << outcome.GetError().GetMessage()
                      << std::endl;
            return false;
        }
```

- Para obtener más información sobre la API, consulta DescribeAutoScalingInstancesla Referencia AWS SDK para C++ de la API.

CLI

AWS CLI

Ejemplo 1: Descripción de una o varias instancias

En este ejemplo, se describe la instancia especificada.

```
aws autoscaling describe-auto-scaling-instances \
    --instance-ids i-06905f55584de02da
```

Salida:

```
{
    "AutoScalingInstances": [
        {
            "InstanceId": "i-06905f55584de02da",
            "InstanceType": "t2.micro",
            "AutoScalingGroupName": "my-asg",
            "AvailabilityZone": "us-west-2b",
            "LifecycleState": "InService",
            "HealthStatus": "HEALTHY",
            "ProtectedFromScaleIn": false,
            "LaunchTemplate": {
                "LaunchTemplateId": "lt-1234567890abcde12",
                "LaunchTemplateName": "my-launch-template",
                "Version": "1"
            }
        }
    ]
}
```

Ejemplo 2: Descripción de una o varias instancias

En este ejemplo, se utiliza la opción `--max-items` para especificar cuántas instancias se devolverán con esta llamada.

```
aws autoscaling describe-auto-scaling-instances \
    --max-items 1
```

Si la salida incluye un campo `NextToken`, hay más instancias. Para obtener las instancias adicionales, utilice el valor de este campo con la opción `--starting-token` en una llamada posterior de la siguiente manera.

```
aws autoscaling describe-auto-scaling-instances \
    --starting-token Z3M3LMPEXAMPLE
```

Consulte el ejemplo 1 para ver una salida de muestra.

Ejemplo 3: cómo describir las instancias que utilizan configuraciones de inicialización

En este ejemplo, se usa la opción `--query` para describir las instancias que usan configuraciones de inicialización.

```
aws autoscaling describe-auto-scaling-instances \
    --query 'AutoScalingInstances[?LaunchConfigurationName!=`null`]'
```

Salida:

```
[
    {
        "InstanceId": "i-088c57934a6449037",
        "InstanceType": "t2.micro",
        "AutoScalingGroupName": "my-asg",
        "AvailabilityZone": "us-west-2c",
        "LifecycleState": "InService",
        "HealthStatus": "HEALTHY",
        "LaunchConfigurationName": "my-lc",
        "ProtectedFromScaleIn": false
    }
]
```

Para obtener más información, consulte Filtrar la salida AWS CLI en la Guía del usuario de la interfaz de línea de AWS comandos.

- Para obtener más información sobre la API, consulte DescribeAutoScalingInstancesla 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 describeAutoScalingInstance(AutoScalingClient
autoScalingClient, String id) {
       try {
           DescribeAutoScalingInstancesRequest
describeAutoScalingInstancesRequest = DescribeAutoScalingInstancesRequest
                   .builder()
                   .instanceIds(id)
                   .build();

           DescribeAutoScalingInstancesResponse response = autoScalingClient

.describeAutoScalingInstances(describeAutoScalingInstancesRequest);
           List<AutoScalingInstanceDetails> instances =
response.autoScalingInstances();
           for (AutoScalingInstanceDetails instance : instances) {
               System.out.println("The instance lifecycle state is: " +
instance.lifecycleState());
           }

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

- Para obtener más información sobre la API, consulta DescribeAutoScalingInstancesla Referencia AWS SDK for Java 2.x 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 describeAutoScalingInstance(id: String) {
    val describeAutoScalingInstancesRequest =
        DescribeAutoScalingInstancesRequest {
            instanceIds = listOf(id)
        }

    AutoScalingClient { region = "us-east-1" }.use { autoScalingClient ->
        val response =
 autoScalingClient.describeAutoScalingInstances(describeAutoScalingInstancesRequest)
        response.autoScalingInstances?.forEach { group ->
            println("The instance lifecycle state is: ${group.lifecycleState}")
        }
    }
}
```

- Para obtener más información sobre la API, consulta DescribeAutoScalingInstancesla referencia sobre el AWS SDK para la API de Kotlin.

PHP

SDK para PHP

> **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 function describeAutoScalingInstances($instanceIds)
```

```
    {
        return $this->autoScalingClient->describeAutoScalingInstances([
            'InstanceIds' => $instanceIds
        ]);
    }
```

- Para obtener más información sobre la API, consulta DescribeAutoScalingInstancesla Referencia AWS SDK para PHP de la API.

PowerShell

Herramientas para la PowerShell versión 4

Ejemplo 1: En este ejemplo se enumeran las instancias IDs de Auto Scaling.

```
Get-ASAutoScalingInstance | format-table -property InstanceId
```

Salida:

```
InstanceId
----------
i-12345678
i-87654321
i-abcd1234
```

Ejemplo 2: En este ejemplo se describe la instancia de escalado automático especificada.

```
Get-ASAutoScalingInstance -InstanceId i-12345678
```

Salida:

```
AutoScalingGroupName    : my-asg
AvailabilityZone        : us-west-2b
HealthStatus            : HEALTHY
InstanceId              : i-12345678
LaunchConfigurationName : my-lc
LifecycleState          : InService
```

Ejemplo 3: En este ejemplo se describen las dos instancias de escalado automático especificadas.

```
Get-ASAutoScalingInstance -InstanceId @("i-12345678", "i-87654321")
```

Ejemplo 4: En este ejemplo, se describen las instancias de escalado automático para el grupo de escalado automático especificado.

```
(Get-ASAutoScalingGroup -AutoScalingGroupName my-asg).Instances | Get-
ASAutoScalingInstance
```

Ejemplo 5: En este ejemplo se describen todas las instancias de escalado automático.

```
Get-ASAutoScalingInstance
```

- Para obtener más información sobre la API, consulte DescribeAutoScalingInstances Herramientas de AWS para PowerShellCmdlet Reference (V4).

Herramientas para la versión 5 PowerShell

Ejemplo 1: En este ejemplo se enumeran las instancias IDs de Auto Scaling.

```
Get-ASAutoScalingInstance | format-table -property InstanceId
```

Salida:

```
InstanceId
----------
i-12345678
i-87654321
i-abcd1234
```

Ejemplo 2: En este ejemplo se describe la instancia de escalado automático especificada.

```
Get-ASAutoScalingInstance -InstanceId i-12345678
```

Salida:

```
AutoScalingGroupName    : my-asg
AvailabilityZone        : us-west-2b
HealthStatus            : HEALTHY
```

```
InstanceId              : i-12345678
LaunchConfigurationName : my-lc
LifecycleState          : InService
```

Ejemplo 3: En este ejemplo se describen las dos instancias de escalado automático especificadas.

```
Get-ASAutoScalingInstance -InstanceId @("i-12345678", "i-87654321")
```

Ejemplo 4: En este ejemplo, se describen las instancias de escalado automático para el grupo de escalado automático especificado.

```
(Get-ASAutoScalingGroup -AutoScalingGroupName my-asg).Instances | Get-
ASAutoScalingInstance
```

Ejemplo 5: En este ejemplo se describen todas las instancias de escalado automático.

```
Get-ASAutoScalingInstance
```

- Para obtener más información sobre la API, consulte DescribeAutoScalingInstances Herramientas de AWS para PowerShellCmdlet Reference (V5).

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 AutoScalingWrapper:
    """Encapsulates Amazon EC2 Auto Scaling actions."""

    def __init__(self, autoscaling_client):
        """
        :param autoscaling_client: A Boto3 Amazon EC2 Auto Scaling client.
```

```
        """
        self.autoscaling_client = autoscaling_client


    def describe_instances(self, instance_ids: List[str]) -> List[Dict[str,
Any]]:
        """
        Gets information about instances.

        :param instance_ids: A list of instance IDs to look up.
        :return: A list of dictionaries with information about each instance,
                 or an empty list if none are found.
        :raises ClientError: If there is an error describing the instances.
        """
        try:
            paginator = self.autoscaling_client.get_paginator(
                "describe_auto_scaling_instances"
            )
            response_iterator = paginator.paginate(InstanceIds=instance_ids)

            instances = []
            for response in response_iterator:
                instances.extend(response.get("AutoScalingInstances", []))

            logger.info(f"Successfully described instances: {instance_ids}")

        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            logger.error(
                f"Couldn't describe instances {instance_ids}. Error code:
{error_code}, Message: {err.response['Error']['Message']}"
            )
            raise
        else:
            return instances
```

- Para obtener más información sobre la API, consulta DescribeAutoScalingInstancesla 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.

```
    pub async fn list_instances(&self) -> Result<Vec<String>, ScenarioError> {
        // The direct way to list instances is by using
 DescribeAutoScalingGroup's instances property. However, this returns a
 Vec<Instance>, as opposed to a Vec<AutoScalingInstanceDetails>.
        // Ok(self.get_group().await?.instances.unwrap_or_default().map(|
i| i.instance_id.clone().unwrap_or_default()).filter(|id| !
id.is_empty()).collect())

        // Alternatively, and for the sake of example,
 DescribeAutoScalingInstances returns a list that can be filtered by the client.
        self.autoscaling
            .describe_auto_scaling_instances()
            .into_paginator()
            .items()
            .send()
            .try_collect()
            .await
            .map(|items| {
                items
                    .into_iter()
                    .filter(|i| {
                        i.auto_scaling_group_name.as_deref()
                            == Some(self.auto_scaling_group_name.as_str())
                    })
                    .map(|i| i.instance_id.unwrap_or_default())
                    .filter(|id| !id.is_empty())
                    .collect::<Vec<String>>()
            })
            .map_err(|err| ScenarioError::new("Failed to get list of auto scaling
 instances", &err))
    }
```

- Para obtener más información sobre la API, consulta DescribeAutoScalingInstancesla referencia sobre la API de AWS SDK para Rust.

SAP ABAP

SDK para 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.

```
    " Example: it_instance_ids contains a list of instance IDs

    TRY.
        DATA(lo_output) = ao_asc->describeautoscalinginstances(
          it_instanceids = it_instance_ids ).

        ot_output = lo_output->get_autoscalinginstances( ).

        MESSAGE 'Auto Scaling instances information retrieved successfully' TYPE
'I'.

      CATCH /aws1/cx_ascresrccontionfault INTO DATA(lo_contention).
        RAISE EXCEPTION lo_contention.
      CATCH /aws1/cx_rt_generic INTO DATA(lo_generic_exception).
        RAISE EXCEPTION lo_generic_exception.
    ENDTRY.
```

- Para obtener más información sobre la API, consulte DescribeAutoScalingInstancesla 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, consulteUtilización de este servicio con un AWS SDK. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Utilizar **`DescribeAutoScalingNotificationTypes`** con una CLI

Los siguientes ejemplos de código muestran cómo utilizar `DescribeAutoScalingNotificationTypes`.

CLI

AWS CLI

Cómo describir los tipos de notificaciones disponibles

En este ejemplo se describen los tipos de notificaciones disponibles.

```
aws autoscaling describe-auto-scaling-notification-types
```

Salida:

```
{
    "AutoScalingNotificationTypes": [
        "autoscaling:EC2_INSTANCE_LAUNCH",
        "autoscaling:EC2_INSTANCE_LAUNCH_ERROR",
        "autoscaling:EC2_INSTANCE_TERMINATE",
        "autoscaling:EC2_INSTANCE_TERMINATE_ERROR",
        "autoscaling:TEST_NOTIFICATION"
    ]
}
```

Para obtener más información, consulte Obtener notificaciones de Amazon SNS cuando su grupo de Auto Scaling escala en la Guía del usuario de Amazon EC2 Auto Scaling.

- Para obtener más información sobre la API, consulte DescribeAutoScalingNotificationTypesla Referencia de AWS CLI comandos.

PowerShell

Herramientas para la PowerShell versión 4

Ejemplo 1: En este ejemplo se enumeran los tipos de notificaciones que son compatibles con el escalado automático.

```
Get-ASAutoScalingNotificationType
```

Salida:

```
autoscaling:EC2_INSTANCE_LAUNCH
autoscaling:EC2_INSTANCE_LAUNCH_ERROR
autoscaling:EC2_INSTANCE_TERMINATE
autoscaling:EC2_INSTANCE_TERMINATE_ERROR
autoscaling:TEST_NOTIFICATION
```

- Para obtener más información sobre la API, consulte DescribeAutoScalingNotificationTypes Herramientas de AWS para PowerShellCmdlet Reference (V4).

Herramientas para la versión 5 PowerShell

Ejemplo 1: En este ejemplo se enumeran los tipos de notificaciones que son compatibles con el escalado automático.

```
Get-ASAutoScalingNotificationType
```

Salida:

```
autoscaling:EC2_INSTANCE_LAUNCH
autoscaling:EC2_INSTANCE_LAUNCH_ERROR
autoscaling:EC2_INSTANCE_TERMINATE
autoscaling:EC2_INSTANCE_TERMINATE_ERROR
autoscaling:TEST_NOTIFICATION
```

- Para obtener más información sobre la API, consulte DescribeAutoScalingNotificationTypesla referencia de Herramientas de AWS para PowerShell cmdlets (V5).

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. Utilización de este servicio con un AWS SDK En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Utilizar **`DescribeLaunchConfigurations`** con una CLI

Los siguientes ejemplos de código muestran cómo utilizar `DescribeLaunchConfigurations`.

CLI

AWS CLI

Ejemplo 1: Cómo describir la configuración de inicialización especificada

En este ejemplo, se describe la configuración de inicialización especificada.

```
aws autoscaling describe-launch-configurations \
    --launch-configuration-names my-launch-config
```

Salida:

```
{
    "LaunchConfigurations": [
        {
            "LaunchConfigurationName": "my-launch-config",
            "LaunchConfigurationARN": "arn:aws:autoscaling:us-
west-2:123456789012:launchConfiguration:98d3b196-4cf9-4e88-8ca1-8547c24ced8b:launchConfig
my-launch-config",
            "ImageId": "ami-0528a5175983e7f28",
            "KeyName": "my-key-pair-uswest2",
            "SecurityGroups": [
                "sg-05eaec502fcdadc2e"
            ],
            "ClassicLinkVPCSecurityGroups": [],
            "UserData": "",
            "InstanceType": "t2.micro",
            "KernelId": "",
            "RamdiskId": "",
            "BlockDeviceMappings": [
                {
                    "DeviceName": "/dev/xvda",
                    "Ebs": {
                        "SnapshotId": "snap-06c1606ba5ca274b1",
                        "VolumeSize": 8,
                        "VolumeType": "gp2",
                        "DeleteOnTermination": true,
                        "Encrypted": false
                    }
                }
            ],
            "InstanceMonitoring": {
```

```
                "Enabled": true
            },
            "CreatedTime": "2020-10-28T02:39:22.321Z",
            "EbsOptimized": false,
            "AssociatePublicIpAddress": true,
            "MetadataOptions": {
                "HttpTokens": "required",
                "HttpPutResponseHopLimit": 1,
                "HttpEndpoint": "disabled"
            }
        }
    ]
}
```

Ejemplo 2: describir un número específico de configuraciones de inicialización

Para devolver un número específico de configuraciones de inicialización, utilice la opción `--max-items`.

```
aws autoscaling describe-launch-configurations \
    --max-items 1
```

Si la salida incluye un campo de `NextToken`, hay más configuraciones de inicialización. Para obtener las configuraciones adicionales de inicialización, utilice el valor de este campo con la opción `--starting-token` en una llamada posterior de la siguiente manera.

```
aws autoscaling describe-launch-configurations \
    --starting-token Z3M3LMPEXAMPLE
```

- Para obtener más información sobre la API, consulte DescribeLaunchConfigurationsla Referencia de AWS CLI comandos.

PowerShell

Herramientas para la PowerShell versión 4

Ejemplo 1: En este ejemplo se muestran los nombres de las configuraciones de inicialización.

```
Get-ASLaunchConfiguration | format-table -property LaunchConfigurationName
```

Salida:

```
LaunchConfigurationName
-----------------------
my-lc-1
my-lc-2
my-lc-3
my-lc-4
my-lc-5
```

Ejemplo 2: En este ejemplo se describe la configuración de inicialización especificada.

```
Get-ASLaunchConfiguration -LaunchConfigurationName my-lc-1
```

Salida:

```
AssociatePublicIpAddress     : True
BlockDeviceMappings          : {/dev/xvda}
ClassicLinkVPCId             :
ClassicLinkVPCSecurityGroups : {}
CreatedTime                  : 12/12/2014 3:22:08 PM
EbsOptimized                 : False
IamInstanceProfile           :
ImageId                      : ami-043a5034
InstanceMonitoring           : Amazon.AutoScaling.Model.InstanceMonitoring
InstanceType                 : t2.micro
KernelId                     :
KeyName                      :
LaunchConfigurationARN       : arn:aws:autoscaling:us-
west-2:123456789012:launchConfiguration:7e5f31e4-693b-4604-9322-
                               e6f68d7fafad:launchConfigurationName/my-lc-1
LaunchConfigurationName      : my-lc-1
PlacementTenancy             :
RamdiskId                    :
SecurityGroups               : {sg-67ef0308}
SpotPrice                    :
UserData                     :
```

Ejemplo 3: En este ejemplo se describen las dos configuraciones de inicialización especificadas.

```
Get-ASLaunchConfiguration -LaunchConfigurationName @("my-lc-1", "my-lc-2")
```

Ejemplo 4: En este ejemplo se describen todas las configuraciones de inicialización.

```
Get-ASLaunchConfiguration
```

- Para obtener más información sobre la API, consulte DescribeLaunchConfigurations Herramientas de AWS para PowerShellCmdlet Reference (V4).

Herramientas para la versión 5 PowerShell

Ejemplo 1: En este ejemplo se muestran los nombres de las configuraciones de inicialización.

```
Get-ASLaunchConfiguration | format-table -property LaunchConfigurationName
```

Salida:

```
LaunchConfigurationName
-----------------------
my-lc-1
my-lc-2
my-lc-3
my-lc-4
my-lc-5
```

Ejemplo 2: En este ejemplo se describe la configuración de inicialización especificada.

```
Get-ASLaunchConfiguration -LaunchConfigurationName my-lc-1
```

Salida:

```
AssociatePublicIpAddress     : True
BlockDeviceMappings          : {/dev/xvda}
ClassicLinkVPCId             :
ClassicLinkVPCSecurityGroups : {}
CreatedTime                  : 12/12/2014 3:22:08 PM
EbsOptimized                 : False
IamInstanceProfile           :
ImageId                      : ami-043a5034
InstanceMonitoring           : Amazon.AutoScaling.Model.InstanceMonitoring
InstanceType                 : t2.micro
KernelId                     :
KeyName                      :
```

```
LaunchConfigurationARN       : arn:aws:autoscaling:us-
west-2:123456789012:launchConfiguration:7e5f31e4-693b-4604-9322-
                               e6f68d7fafad:launchConfigurationName/my-lc-1
LaunchConfigurationName      : my-lc-1
PlacementTenancy             :
RamdiskId                    :
SecurityGroups               : {sg-67ef0308}
SpotPrice                    :
UserData                     :
```

Ejemplo 3: En este ejemplo se describen las dos configuraciones de inicialización especificadas.

```
Get-ASLaunchConfiguration -LaunchConfigurationName @("my-lc-1", "my-lc-2")
```

Ejemplo 4: En este ejemplo se describen todas las configuraciones de inicialización.

```
Get-ASLaunchConfiguration
```

- Para obtener más información sobre la API, consulte DescribeLaunchConfigurationsla referencia de Herramientas de AWS para PowerShell cmdlets (V5).

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. Utilización de este servicio con un AWS SDK En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Utilizar **`DescribeLifecycleHookTypes`** con una CLI

Los siguientes ejemplos de código muestran cómo utilizar `DescribeLifecycleHookTypes`.

CLI

AWS CLI

Cómo describir los tipos disponibles de enlaces de ciclo de vida

En este ejemplo, se describen los tipos disponibles de enlaces de ciclo de vida.

```
aws autoscaling describe-lifecycle-hook-types
```

Salida:

```
{
    "LifecycleHookTypes": [
        "autoscaling:EC2_INSTANCE_LAUNCHING",
        "autoscaling:EC2_INSTANCE_TERMINATING"
    ]
}
```

- Para obtener más información sobre la API, consulte DescribeLifecycleHookTypesla Referencia de AWS CLI comandos.

PowerShell

Herramientas para la PowerShell versión 4

Ejemplo 1: En este ejemplo se enumeran los tipos de enlaces de ciclo de vida compatibles con el escalado automático.

```
Get-ASLifecycleHookType
```

Salida:

```
autoscaling:EC2_INSTANCE_LAUNCHING
auto-scaling:EC2_INSTANCE_TERMINATING
```

- Para obtener más información sobre la API, consulte DescribeLifecycleHookTypes Herramientas de AWS para PowerShellCmdlet Reference (V4).

Herramientas para la versión 5 PowerShell

Ejemplo 1: En este ejemplo se enumeran los tipos de enlaces de ciclo de vida compatibles con el escalado automático.

```
Get-ASLifecycleHookType
```

Salida:

```
autoscaling:EC2_INSTANCE_LAUNCHING
auto-scaling:EC2_INSTANCE_TERMINATING
```

- Para obtener más información sobre la API, consulte DescribeLifecycleHookTypesla referencia de Herramientas de AWS para PowerShell cmdlets (V5).

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. Utilización de este servicio con un AWS SDK En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Utilizar **`DescribeLifecycleHooks`** con una CLI

Los siguientes ejemplos de código muestran cómo utilizar `DescribeLifecycleHooks`.

CLI

AWS CLI

Cómo describir los enlaces de ciclo de vida

En este ejemplo, se describen los enlaces de ciclo de vida para el grupo de escalado automático especificado.

```
aws autoscaling describe-lifecycle-hooks \
    --auto-scaling-group-name my-asg
```

Salida:

```
{
    "LifecycleHooks": [
        {
            "GlobalTimeout": 3000,
            "HeartbeatTimeout": 30,
            "AutoScalingGroupName": "my-asg",
            "LifecycleHookName": "my-launch-hook",
            "DefaultResult": "ABANDON",
            "LifecycleTransition": "autoscaling:EC2_INSTANCE_LAUNCHING"
        },
        {
            "GlobalTimeout": 6000,
            "HeartbeatTimeout": 60,
            "AutoScalingGroupName": "my-asg",
            "LifecycleHookName": "my-termination-hook",
            "DefaultResult": "CONTINUE",
            "LifecycleTransition": "autoscaling:EC2_INSTANCE_TERMINATING"
```

```
        }
    ]
}
```

- Para obtener más información sobre la API, consulte DescribeLifecycleHooksla Referencia de AWS CLI comandos.

PowerShell

Herramientas para la PowerShell versión 4

Ejemplo 1: En este ejemplo se describe el enlace de ciclo de vida especificado.

```
Get-ASLifecycleHook -AutoScalingGroupName my-asg -LifecycleHookName
 myLifecycleHook
```

Salida:

```
AutoScalingGroupName  : my-asg
DefaultResult         : ABANDON
GlobalTimeout         : 172800
HeartbeatTimeout      : 3600
LifecycleHookName     : myLifecycleHook
LifecycleTransition   : auto-scaling:EC2_INSTANCE_LAUNCHING
NotificationMetadata  :
NotificationTargetARN : arn:aws:sns:us-west-2:123456789012:my-topic
RoleARN               : arn:aws:iam::123456789012:role/my-iam-role
```

Ejemplo 2: En este ejemplo se describen los enlaces de ciclo de vida del grupo de escalado automático especificado.

```
Get-ASLifecycleHook -AutoScalingGroupName my-asg
```

Ejemplo 3: En este ejemplo se describen todos los enlaces de ciclo de vida para todos los grupos de escalado automático.

```
Get-ASLifecycleHook
```

- Para obtener más información sobre la API, consulte DescribeLifecycleHooks Herramientas de AWS para PowerShellCmdlet Reference (V4).

Herramientas para la versión 5 PowerShell

Ejemplo 1: En este ejemplo se describe el enlace de ciclo de vida especificado.

```
Get-ASLifecycleHook -AutoScalingGroupName my-asg -LifecycleHookName
 myLifecycleHook
```

Salida:

```
AutoScalingGroupName  : my-asg
DefaultResult         : ABANDON
GlobalTimeout         : 172800
HeartbeatTimeout      : 3600
LifecycleHookName     : myLifecycleHook
LifecycleTransition   : auto-scaling:EC2_INSTANCE_LAUNCHING
NotificationMetadata  :
NotificationTargetARN : arn:aws:sns:us-west-2:123456789012:my-topic
RoleARN               : arn:aws:iam::123456789012:role/my-iam-role
```

Ejemplo 2: En este ejemplo se describen los enlaces de ciclo de vida del grupo de escalado automático especificado.

```
Get-ASLifecycleHook -AutoScalingGroupName my-asg
```

Ejemplo 3: En este ejemplo se describen todos los enlaces de ciclo de vida para todos los grupos de escalado automático.

```
Get-ASLifecycleHook
```

- Para obtener más información sobre la API, consulte DescribeLifecycleHooksla referencia de Herramientas de AWS para PowerShell cmdlets (V5).

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. Utilización de este servicio con un AWS SDK En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Utilizar **`DescribeLoadBalancers`** con una CLI

Los siguientes ejemplos de código muestran cómo utilizar `DescribeLoadBalancers`.

CLI

AWS CLI

Cómo describir los equilibradores de carga clásicos para un grupo de escalado automático

En este ejemplo se describen los equilibradores de carga clásicos del grupo de escalado automático especificado.

```
aws autoscaling describe-load-balancers \
    --auto-scaling-group-name my-asg
```

Salida:

```
{
    "LoadBalancers": [
        {
            "State": "Added",
            "LoadBalancerName": "my-load-balancer"
        }
    ]
}
```

- Para obtener más información sobre la API, consulte DescribeLoadBalancersla Referencia de AWS CLI comandos.

PowerShell

Herramientas para la PowerShell versión 4

Ejemplo 1: En este ejemplo se describen los equilibradores de carga del grupo de escalado automático especificado.

```
Get-ASLoadBalancer -AutoScalingGroupName my-asg
```

Salida:

```
LoadBalancerName    State
----------------    -----
my-lb               Added
```

- Para obtener más información sobre la API, consulte DescribeLoadBalancers Herramientas de AWS para PowerShellCmdlet Reference (V4).

Herramientas para la versión 5 PowerShell

Ejemplo 1: En este ejemplo se describen los equilibradores de carga del grupo de escalado automático especificado.

```
Get-ASLoadBalancer -AutoScalingGroupName my-asg
```

Salida:

```
LoadBalancerName    State
----------------    -----
my-lb               Added
```

- Para obtener más información sobre la API, consulte DescribeLoadBalancersla referencia de Herramientas de AWS para PowerShell cmdlets (V5).

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. Utilización de este servicio con un AWS SDK En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Utilizar **`DescribeMetricCollectionTypes`** con una CLI

Los siguientes ejemplos de código muestran cómo utilizar `DescribeMetricCollectionTypes`.

CLI

AWS CLI

Cómo describir los tipos de recopilación de métricas disponibles

En este ejemplo, se describen los tipos de recopilación de métricas disponibles.

```
aws autoscaling describe-metric-collection-types
```

Salida:

```
{
    "Metrics": [
        {
```

```
            "Metric": "GroupMinSize"
        },
        {
            "Metric": "GroupMaxSize"
        },
        {
            "Metric": "GroupDesiredCapacity"
        },
        {
            "Metric": "GroupInServiceInstances"
        },
        {
            "Metric": "GroupInServiceCapacity"
        },
        {
            "Metric": "GroupPendingInstances"
        },
        {
            "Metric": "GroupPendingCapacity"
        },
        {
            "Metric": "GroupTerminatingInstances"
        },
        {
            "Metric": "GroupTerminatingCapacity"
        },
        {
            "Metric": "GroupStandbyInstances"
        },
        {
            "Metric": "GroupStandbyCapacity"
        },
        {
            "Metric": "GroupTotalInstances"
        },
        {
            "Metric": "GroupTotalCapacity"
        }
    ],
    "Granularities": [
        {
            "Granularity": "1Minute"
        }
    ]
```

```
}
```

Para obtener más información, consulte Métricas de grupo de Auto Scaling en la Guía del usuario de Amazon EC2 Auto Scaling.

- Para obtener más información sobre la API, consulte DescribeMetricCollectionTypesla Referencia de AWS CLI comandos.

PowerShell

Herramientas para la PowerShell versión 4

Ejemplo 1: En este ejemplo se enumeran los tipos de recopilación de métricas que admite el escalado automático.

```
(Get-ASMetricCollectionType).Metrics
```

Salida:

```
Metric
------
GroupMinSize
GroupMaxSize
GroupDesiredCapacity
GroupInServiceInstances
GroupPendingInstances
GroupTerminatingInstances
GroupStandbyInstances
GroupTotalInstances
```

Ejemplo 2: En este ejemplo se enumeran las granularidades correspondientes.

```
(Get-ASMetricCollectionType).Granularities
```

Salida:

```
Granularity
-----------
1Minute
```

- Para obtener más información sobre la API, consulte DescribeMetricCollectionTypes Herramientas de AWS para PowerShellCmdlet Reference (V4).

Herramientas para la versión 5 PowerShell

Ejemplo 1: En este ejemplo se enumeran los tipos de recopilación de métricas que admite el escalado automático.

```
(Get-ASMetricCollectionType).Metrics
```

Salida:

```
Metric
------
GroupMinSize
GroupMaxSize
GroupDesiredCapacity
GroupInServiceInstances
GroupPendingInstances
GroupTerminatingInstances
GroupStandbyInstances
GroupTotalInstances
```

Ejemplo 2: En este ejemplo se enumeran las granularidades correspondientes.

```
(Get-ASMetricCollectionType).Granularities
```

Salida:

```
Granularity
-----------
1Minute
```

- Para obtener más información sobre la API, consulte DescribeMetricCollectionTypesla referencia de Herramientas de AWS para PowerShell cmdlets (V5).

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. Utilización de este servicio con un AWS SDK En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Utilizar **`DescribeNotificationConfigurations`** con una CLI

Los siguientes ejemplos de código muestran cómo utilizar `DescribeNotificationConfigurations`.

CLI

AWS CLI

Ejemplo 1: Cómo describir las configuraciones de notificación de un grupo especificado

En este ejemplo, se describen las configuraciones de notificación para el grupo de escalado automático especificado.

```
aws autoscaling describe-notification-configurations \
    --auto-scaling-group-name my-asg
```

Salida:

```
{
    "NotificationConfigurations": [
        {
            "AutoScalingGroupName": "my-asg",
            "NotificationType": "autoscaling:TEST_NOTIFICATION",
            "TopicARN": "arn:aws:sns:us-west-2:123456789012:my-sns-topic-2"
        },
        {
            "AutoScalingGroupName": "my-asg",
            "NotificationType": "autoscaling:TEST_NOTIFICATION",
            "TopicARN": "arn:aws:sns:us-west-2:123456789012:my-sns-topic"
        }
    ]
}
```

Para obtener más información, consulte Obtener notificaciones de Amazon SNS cuando su grupo de Auto Scaling escala en la Guía del usuario de Amazon EC2 Auto Scaling.

Ejemplo 1: describir un número específico de configuraciones de notificaciones

Para devolver un número específico de configuraciones de notificación, utilice el parámetro `max-items`.

```
aws autoscaling describe-notification-configurations \
    --auto-scaling-group-name my-auto-scaling-group \
    --max-items 1
```

Salida:

```
{
    "NotificationConfigurations": [
        {
            "AutoScalingGroupName": "my-asg",
            "NotificationType": "autoscaling:TEST_NOTIFICATION",
            "TopicARN": "arn:aws:sns:us-west-2:123456789012:my-sns-topic-2"
        },
        {
            "AutoScalingGroupName": "my-asg",
            "NotificationType": "autoscaling:TEST_NOTIFICATION",
            "TopicARN": "arn:aws:sns:us-west-2:123456789012:my-sns-topic"
        }
    ]
}
```

Si la salida incluye un campo `NextToken` de, hay más configuraciones de notificación. Para obtener las configuraciones de notificación adicionales, utilice el valor de este campo con el parámetro `starting-token` en una llamada posterior de la siguiente manera.

```
aws autoscaling describe-notification-configurations \
    --auto-scaling-group-name my-asg \
    --starting-token Z3M3LMPEXAMPLE
```

Para obtener más información, consulte Obtener notificaciones de Amazon SNS cuando su grupo de Auto Scaling escala en la Guía del usuario de Amazon EC2 Auto Scaling.

- Para obtener más información sobre la API, consulte DescribeNotificationConfigurationsla Referencia de AWS CLI comandos.

PowerShell

Herramientas para la PowerShell versión 4

Ejemplo 1: En este ejemplo se describen las acciones de notificación asociadas al grupo de escalado automático especificado.

```
Get-ASNotificationConfiguration -AutoScalingGroupName my-asg | format-list
```

Salida:

```
AutoScalingGroupName : my-asg
NotificationType     : auto-scaling:EC2_INSTANCE_LAUNCH
TopicARN             : arn:aws:sns:us-west-2:123456789012:my-topic

AutoScalingGroupName : my-asg
NotificationType     : auto-scaling:EC2_INSTANCE_TERMINATE
TopicARN             : arn:aws:sns:us-west-2:123456789012:my-topic
```

Ejemplo 2: En este ejemplo se describen las acciones de notificación asociadas a todos los grupos de escalado automático.

```
Get-ASNotificationConfiguration
```

- Para obtener más información sobre la API, consulte DescribeNotificationConfigurations Herramientas de AWS para PowerShellCmdlet Reference (V4).

Herramientas para la versión 5 PowerShell

Ejemplo 1: En este ejemplo se describen las acciones de notificación asociadas al grupo de escalado automático especificado.

```
Get-ASNotificationConfiguration -AutoScalingGroupName my-asg | format-list
```

Salida:

```
AutoScalingGroupName : my-asg
NotificationType     : auto-scaling:EC2_INSTANCE_LAUNCH
TopicARN             : arn:aws:sns:us-west-2:123456789012:my-topic

AutoScalingGroupName : my-asg
NotificationType     : auto-scaling:EC2_INSTANCE_TERMINATE
TopicARN             : arn:aws:sns:us-west-2:123456789012:my-topic
```

Ejemplo 2: En este ejemplo se describen las acciones de notificación asociadas a todos los grupos de escalado automático.

```
Get-ASNotificationConfiguration
```

- Para obtener más información sobre la API, consulte DescribeNotificationConfigurationsla referencia de Herramientas de AWS para PowerShell cmdlets (V5).

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. Utilización de este servicio con un AWS SDK En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Utilizar **`DescribePolicies`** con una CLI

Los siguientes ejemplos de código muestran cómo utilizar `DescribePolicies`.

CLI

AWS CLI

Ejemplo 1: Cómo describir las políticas de escalado del grupo especificado

En este ejemplo, se describen las políticas de escalado para el grupo de escalado automático especificado.

```
aws autoscaling describe-policies \
    --auto-scaling-group-name my-asg
```

Salida:

```
{
    "ScalingPolicies": [
        {
            "AutoScalingGroupName": "my-asg",
            "PolicyName": "alb1000-target-tracking-scaling-policy",
            "PolicyARN": "arn:aws:autoscaling:us-
west-2:123456789012:scalingPolicy:3065d9c8-9969-4bec-
bb6a-3fbe5550fde6:autoScalingGroupName/my-asg:policyName/alb1000-target-tracking-
scaling-policy",
            "PolicyType": "TargetTrackingScaling",
            "StepAdjustments": [],
            "Alarms": [
                {
```

```
                    "AlarmName": "TargetTracking-my-asg-
AlarmHigh-924887a9-12d7-4e01-8686-6f844d13a196",
                    "AlarmARN": "arn:aws:cloudwatch:us-
west-2:123456789012:alarm:TargetTracking-my-asg-
AlarmHigh-924887a9-12d7-4e01-8686-6f844d13a196"
                },
                {
                    "AlarmName": "TargetTracking-my-asg-AlarmLow-f96f899d-
b8e7-4d09-a010-c1aaa35da296",
                    "AlarmARN": "arn:aws:cloudwatch:us-
west-2:123456789012:alarm:TargetTracking-my-asg-AlarmLow-f96f899d-b8e7-4d09-a010-
c1aaa35da296"
                }
            ],
            "TargetTrackingConfiguration": {
                "PredefinedMetricSpecification": {
                    "PredefinedMetricType": "ALBRequestCountPerTarget",
                    "ResourceLabel": "app/my-alb/778d41231b141a0f/targetgroup/my-
alb-target-group/943f017f100becff"
                },
                "TargetValue": 1000.0,
                "DisableScaleIn": false
            },
            "Enabled": true
        },
        {
            "AutoScalingGroupName": "my-asg",
            "PolicyName": "cpu40-target-tracking-scaling-policy",
            "PolicyARN": "arn:aws:autoscaling:us-
west-2:123456789012:scalingPolicy:5fd26f71-39d4-4690-82a9-
b8515c45cdde:autoScalingGroupName/my-asg:policyName/cpu40-target-tracking-
scaling-policy",
            "PolicyType": "TargetTrackingScaling",
            "StepAdjustments": [],
            "Alarms": [
                {
                    "AlarmName": "TargetTracking-my-asg-
AlarmHigh-139f9789-37b9-42ad-bea5-b5b147d7f473",
                    "AlarmARN": "arn:aws:cloudwatch:us-
west-2:123456789012:alarm:TargetTracking-my-asg-AlarmHigh-139f9789-37b9-42ad-
bea5-b5b147d7f473"
                },
                {
```

```
                    "AlarmName": "TargetTracking-my-asg-AlarmLow-bd681c67-
fc18-4c56-8468-fb8e413009c9",
                    "AlarmARN": "arn:aws:cloudwatch:us-
west-2:123456789012:alarm:TargetTracking-my-asg-AlarmLow-bd681c67-fc18-4c56-8468-
fb8e413009c9"
                }
            ],
            "TargetTrackingConfiguration": {
                "PredefinedMetricSpecification": {
                    "PredefinedMetricType": "ASGAverageCPUUtilization"
                },
                "TargetValue": 40.0,
                "DisableScaleIn": false
            },
            "Enabled": true
        }
    ]
}
```

Para obtener más información, consulte Escalado dinámico en la Guía del usuario de Amazon EC2 Auto Scaling.

Ejemplo 2: describir las políticas de escalado de un nombre especificado

Para devolver políticas de escalado específicas, utilice la opción `--policy-names`.

```
aws autoscaling describe-policies \
    --auto-scaling-group-name my-asg \
    --policy-names cpu40-target-tracking-scaling-policy
```

Consulte el ejemplo 1 para ver una salida de muestra.

Para obtener más información, consulte Escalado dinámico en la Guía del usuario de Amazon EC2 Auto Scaling.

Ejemplo 3: cómo describir una serie de políticas de escalado

Para devolver un número específico de políticas, utilice la opción `--max-items`.

```
aws autoscaling describe-policies \
    --auto-scaling-group-name my-asg \
    --max-items 1
```

Consulte el ejemplo 1 para ver una salida de muestra.

Si la salida incluye un campo de `NextToken`, utilice el valor de este campo con la opción `--starting-token` en una llamada posterior para obtener las políticas adicionales.

```
aws autoscaling describe-policies --auto-scaling-group-name my-asg --starting-token Z3M3LMPEXAMPLE
```

Para obtener más información, consulte Dynamic scaling en la Guía del usuario de Amazon EC2 Auto Scaling.

- Para obtener más información sobre la API, consulte DescribePoliciesla Referencia de AWS CLI comandos.

PowerShell

Herramientas para la PowerShell versión 4

Ejemplo 1: En este ejemplo, se describen todas las políticas para el grupo de escalado automático especificado.

```
Get-ASPolicy -AutoScalingGroupName my-asg
```

Salida:

```
AdjustmentType          : ChangeInCapacity
Alarms                  : {}
AutoScalingGroupName    : my-asg
Cooldown                : 0
EstimatedInstanceWarmup : 0
MetricAggregationType   :
MinAdjustmentMagnitude  : 0
MinAdjustmentStep       : 0
PolicyARN               : arn:aws:auto-scaling:us-
west-2:123456789012:scalingPolicy:aa3836ab-5462-42c7-adab-e1d769fc24ef
                          :autoScalingGroupName/my-asg:policyName/myScaleInPolicy
PolicyName              : myScaleInPolicy
PolicyType              : SimpleScaling
ScalingAdjustment       : -1
StepAdjustments         : {}
```

Ejemplo 2: En este ejemplo se describen las políticas especificadas para el grupo de escalado automático especificado.

```
Get-ASPolicy -AutoScalingGroupName my-asg -PolicyName @("myScaleOutPolicy",
 "myScaleInPolicy")
```

Ejemplo 3: En este ejemplo se describen todas las políticas para todos los grupos de escalado automático.

```
Get-ASPolicy
```

- Para obtener más información sobre la API, consulte DescribePolicies Herramientas de AWS para PowerShellCmdlet Reference (V4).

Herramientas para la versión 5 PowerShell

Ejemplo 1: En este ejemplo, se describen todas las políticas para el grupo de escalado automático especificado.

```
Get-ASPolicy -AutoScalingGroupName my-asg
```

Salida:

```
AdjustmentType          : ChangeInCapacity
Alarms                  : {}
AutoScalingGroupName    : my-asg
Cooldown                : 0
EstimatedInstanceWarmup : 0
MetricAggregationType   :
MinAdjustmentMagnitude  : 0
MinAdjustmentStep       : 0
PolicyARN               : arn:aws:auto-scaling:us-
west-2:123456789012:scalingPolicy:aa3836ab-5462-42c7-adab-e1d769fc24ef
                          :autoScalingGroupName/my-asg:policyName/myScaleInPolicy
PolicyName              : myScaleInPolicy
PolicyType              : SimpleScaling
ScalingAdjustment       : -1
StepAdjustments         : {}
```

Ejemplo 2: En este ejemplo se describen las políticas especificadas para el grupo de escalado automático especificado.

```
Get-ASPolicy -AutoScalingGroupName my-asg -PolicyName @("myScaleOutPolicy",
 "myScaleInPolicy")
```

Ejemplo 3: En este ejemplo se describen todas las políticas para todos los grupos de escalado automático.

```
Get-ASPolicy
```

- Para obtener más información sobre la API, consulte DescribePoliciesla referencia de Herramientas de AWS para PowerShell cmdlets (V5).

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. Utilización de este servicio con un AWS SDK En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Úselo **`DescribeScalingActivities`** con un AWS SDK o CLI

Los siguientes ejemplos de código muestran cómo utilizar `DescribeScalingActivities`.

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:

- Conceptos básicos

.NET

SDK para .NET (v4)

> **Note**
>
> Hay más información GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
    /// <summary>
    /// Retrieve a list of the Amazon EC2 Auto Scaling activities for an
    /// Amazon EC2 Auto Scaling group.
    /// </summary>
```

```
    /// <param name="groupName">The name of the Amazon EC2 Auto Scaling group.</
param>
    /// <returns>A list of Amazon EC2 Auto Scaling activities.</returns>
    public async Task<List<Activity>> DescribeScalingActivitiesAsync(
        string groupName)
    {
        var activities = new List<Activity>();
        var scalingActivitiesRequest = new DescribeScalingActivitiesRequest
        {
            AutoScalingGroupName = groupName,
            MaxRecords = 10,
        };

        var response = await
 _amazonAutoScaling.DescribeScalingActivitiesAsync(scalingActivitiesRequest);
        if (response.Activities != null)
        {
            activities = response.Activities;
        }
        return activities;
    }
```

- Para obtener más información sobre la API, consulta DescribeScalingActivitiesla Referencia AWS SDK para .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::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region (overrides config file).
        // clientConfig.region = "us-east-1";
```

```
    Aws::AutoScaling::AutoScalingClient autoScalingClient(clientConfig);

        Aws::AutoScaling::Model::DescribeScalingActivitiesRequest request;
        request.SetAutoScalingGroupName(groupName);

        Aws::Vector<Aws::AutoScaling::Model::Activity> allActivities;
        Aws::String nextToken; // Used for pagination;
        do {
            if (!nextToken.empty()) {
                request.SetNextToken(nextToken);
            }
            Aws::AutoScaling::Model::DescribeScalingActivitiesOutcome outcome =
                    autoScalingClient.DescribeScalingActivities(request);

            if (outcome.IsSuccess()) {
                const Aws::Vector<Aws::AutoScaling::Model::Activity> &activities
=
                        outcome.GetResult().GetActivities();
                allActivities.insert(allActivities.end(), activities.begin(),
activities.end());
                nextToken  = outcome.GetResult().GetNextToken();
            }
            else {
                std::cerr << "Error with AutoScaling::DescribeScalingActivities.
"
                          << outcome.GetError().GetMessage()
                          << std::endl;

            }
        } while (!nextToken.empty());

        std::cout << "Found " << allActivities.size() << " activities."
                  << std::endl;
        std::cout << "Activities are ordered with the most recent first."
                  << std::endl;
        for (const Aws::AutoScaling::Model::Activity &activity: allActivities) {
            std::cout << activity.GetDescription() << std::endl;
            std::cout << activity.GetDetails() << std::endl;
        }
```

- Para obtener más información sobre la API, consulta DescribeScalingActivitiesla Referencia AWS SDK para C++ de la API.

CLI

AWS CLI

Ejemplo 1: Descripción de las actividades de escalado para el grupo especificado

En este ejemplo, se describen una o varias actividades de escalado para el grupo de escalado automático especificado.

```
aws autoscaling describe-scaling-activities \
    --auto-scaling-group-name my-asg
```

Salida:

```
{
    "Activities": [
        {
            "ActivityId": "f9f2d65b-f1f2-43e7-b46d-d86756459699",
            "Description": "Launching a new EC2 instance: i-0d44425630326060f",
            "AutoScalingGroupName": "my-asg",
            "Cause": "At 2020-10-30T19:35:51Z a user request update of
 AutoScalingGroup constraints to min: 0, max: 16, desired: 16 changing the
 desired capacity from 0 to 16.  At 2020-10-30T19:36:07Z an instance was started
 in response to a difference between desired and actual capacity, increasing the
 capacity from 0 to 16.",
            "StartTime": "2020-10-30T19:36:09.766Z",
            "EndTime": "2020-10-30T19:36:41Z",
            "StatusCode": "Successful",
            "Progress": 100,
            "Details": "{\"Subnet ID\":\"subnet-5ea0c127\",\"Availability Zone\":
\"us-west-2b\"}"
        }
    ]
}
```

Para obtener más información, consulte Verificar una actividad de escalado para un grupo de escalado automático en la Guía del usuario de Amazon EC2 Auto Scaling.

Ejemplo 2: Descripción de las actividades de escalado de un grupo eliminado

Para describir las actividades de escalado después de que un grupo de escalado automático se haya eliminado, añada la opción `--include-deleted-groups`.

```
aws autoscaling describe-scaling-activities \
    --auto-scaling-group-name my-asg \
    --include-deleted-groups
```

Salida:

```
{
    "Activities": [
        {
            "ActivityId": "e1f5de0e-f93e-1417-34ac-092a76fba220",
            "Description": "Launching a new EC2 instance.  Status Reason: Your
 Spot request price of 0.001 is lower than the minimum required Spot request
 fulfillment price of 0.0031. Launching EC2 instance failed.",
            "AutoScalingGroupName": "my-asg",
            "Cause": "At 2021-01-13T20:47:24Z a user request update of
 AutoScalingGroup constraints to min: 1, max: 5, desired: 3 changing the desired
 capacity from 0 to 3.  At 2021-01-13T20:47:27Z an instance was started in
 response to a difference between desired and actual capacity, increasing the
 capacity from 0 to 3.",
            "StartTime": "2021-01-13T20:47:30.094Z",
            "EndTime": "2021-01-13T20:47:30Z",
            "StatusCode": "Failed",
            "StatusMessage": "Your Spot request price of 0.001 is lower than
 the minimum required Spot request fulfillment price of 0.0031. Launching EC2
 instance failed.",
            "Progress": 100,
            "Details": "{\"Subnet ID\":\"subnet-5ea0c127\",\"Availability Zone\":
\"us-west-2b\"}",
            "AutoScalingGroupState": "Deleted",
            "AutoScalingGroupARN": "arn:aws:autoscaling:us-
west-2:123456789012:autoScalingGroup:283179a2-
f3ce-423d-93f6-66bb518232f7:autoScalingGroupName/my-asg"
        }
    ]
}
```

Para obtener más información, consulte Solución de problemas para Amazon EC2 Auto Scaling en la Guía del usuario de Amazon EC2 Auto Scaling.

Ejemplo 3: descripción de un número específico de actividades de escalado

Para devolver un número específico de actividades, utilice la opción `--max-items`.

```
aws autoscaling describe-scaling-activities \
    --max-items 1
```

Salida:

```
{
    "Activities": [
        {
            "ActivityId": "f9f2d65b-f1f2-43e7-b46d-d86756459699",
            "Description": "Launching a new EC2 instance: i-0d44425630326060f",
            "AutoScalingGroupName": "my-asg",
            "Cause": "At 2020-10-30T19:35:51Z a user request update of
 AutoScalingGroup constraints to min: 0, max: 16, desired: 16 changing the
 desired capacity from 0 to 16.  At 2020-10-30T19:36:07Z an instance was started
 in response to a difference between desired and actual capacity, increasing the
 capacity from 0 to 16.",
            "StartTime": "2020-10-30T19:36:09.766Z",
            "EndTime": "2020-10-30T19:36:41Z",
            "StatusCode": "Successful",
            "Progress": 100,
            "Details": "{\"Subnet ID\":\"subnet-5ea0c127\",\"Availability Zone\":
\"us-west-2b\"}"
        }
    ]
}
```

Si la salida incluye un campo `NextToken`, hay más actividades. Para obtener las actividades adicionales, utilice el valor de este campo con la opción `--starting-token` en una llamada posterior de la siguiente manera.

```
aws autoscaling describe-scaling-activities \
    --starting-token Z3M3LMPEXAMPLE
```

Para obtener más información, consulte Verificar una actividad de escalado para un grupo de escalado automático en la Guía del usuario de Amazon EC2 Auto Scaling.

- Para obtener más información sobre la API, consulta DescribeScalingActivitiesla 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 describeScalingActivities(AutoScalingClient
autoScalingClient, String groupName) {
       try {
           DescribeScalingActivitiesRequest scalingActivitiesRequest =
DescribeScalingActivitiesRequest.builder()
                   .autoScalingGroupName(groupName)
                   .maxRecords(10)
                   .build();

           DescribeScalingActivitiesResponse response = autoScalingClient
                   .describeScalingActivities(scalingActivitiesRequest);
           List<Activity> activities = response.activities();
           for (Activity activity : activities) {
               System.out.println("The activity Id is " +
activity.activityId());
               System.out.println("The activity details are " +
activity.details());
           }

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

- Para obtener más información sobre la API, consulta DescribeScalingActivitiesla Referencia AWS SDK for Java 2.x 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 describeAutoScalingGroups(groupName: String) {
    val groupsReques =
        DescribeAutoScalingGroupsRequest {
            autoScalingGroupNames = listOf(groupName)
            maxRecords = 10
        }

    AutoScalingClient { region = "us-east-1" }.use { autoScalingClient ->
        val response = autoScalingClient.describeAutoScalingGroups(groupsReques)
        response.autoScalingGroups?.forEach { group ->
            println("The service to use for the health checks:
 ${group.healthCheckType}")
        }
    }
}
```

- Para obtener más información sobre la API, consulta DescribeScalingActivitiesla referencia sobre el AWS SDK para la API de Kotlin.

PHP

SDK para PHP

> **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 function describeScalingActivities($autoScalingGroupName)
    {
        return $this->autoScalingClient->describeScalingActivities([
            'AutoScalingGroupName' => $autoScalingGroupName,
        ]);
    }
```

- Para obtener más información sobre la API, consulta DescribeScalingActivitiesla Referencia AWS SDK para PHP de la API.

PowerShell

Herramientas para la PowerShell versión 4

Ejemplo 1: En este ejemplo, se describen actividades de escalado de las últimas de seis semanas para el grupo de escalado automático especificado.

```
Get-ASScalingActivity -AutoScalingGroupName my-asg
```

Salida:

```
ActivityId           : 063308ae-aa22-4a9b-94f4-9fae4EXAMPLE
AutoScalingGroupName : my-asg
Cause                : At 2015-11-22T15:45:16Z a user request explicitly set
 group desired capacity changing the desired
                       capacity from 1 to 2.  At 2015-11-22T15:45:34Z an instance
 was started in response to a difference
                       between desired and actual capacity, increasing the
 capacity from 1 to 2.
Description          : Launching a new EC2 instance: i-26e715fc
Details              : {"Availability Zone":"us-west-2b","Subnet
 ID":"subnet-5264e837"}
EndTime              : 11/22/2015 7:46:09 AM
Progress             : 100
StartTime            : 11/22/2015 7:45:35 AM
StatusCode           : Successful
StatusMessage        :

ActivityId           : ce719997-086d-4c73-a2f1-ab703EXAMPLE
AutoScalingGroupName : my-asg
```

```
Cause                : At 2015-11-20T22:57:53Z a user request created an
 AutoScalingGroup changing the desired capacity
                        from 0 to 1.  At 2015-11-20T22:57:58Z an instance was
 started in response to a difference betwe
                       en desired and actual capacity, increasing the capacity
 from 0 to 1.
Description          : Launching a new EC2 instance: i-93633f9b
Details              : {"Availability Zone":"us-west-2b","Subnet
 ID":"subnet-5264e837"}
EndTime              : 11/20/2015 2:58:32 PM
Progress             : 100
StartTime            : 11/20/2015 2:57:59 PM
StatusCode           : Successful
StatusMessage        :
```

Ejemplo 2: En este ejemplo se describe la actividad de escalado especificada.

```
Get-ASScalingActivity -ActivityId "063308ae-aa22-4a9b-94f4-9fae4EXAMPLE"
```

Ejemplo 3: En este ejemplo se describen las actividades de escalado de las últimas seis semanas para todos los grupos de escalado automático.

```
Get-ASScalingActivity
```

- Para obtener más información sobre la API, consulte DescribeScalingActivities Herramientas de AWS para PowerShellCmdlet Reference (V4).

Herramientas para la versión 5 PowerShell

Ejemplo 1: En este ejemplo, se describen actividades de escalado de las últimas de seis semanas para el grupo de escalado automático especificado.

```
Get-ASScalingActivity -AutoScalingGroupName my-asg
```

Salida:

```
ActivityId           : 063308ae-aa22-4a9b-94f4-9fae4EXAMPLE
AutoScalingGroupName : my-asg
Cause                : At 2015-11-22T15:45:16Z a user request explicitly set
 group desired capacity changing the desired
```

```
                      capacity from 1 to 2.  At 2015-11-22T15:45:34Z an instance
 was started in response to a difference
                      between desired and actual capacity, increasing the
 capacity from 1 to 2.
Description          : Launching a new EC2 instance: i-26e715fc
Details              : {"Availability Zone":"us-west-2b","Subnet
 ID":"subnet-5264e837"}
EndTime              : 11/22/2015 7:46:09 AM
Progress             : 100
StartTime            : 11/22/2015 7:45:35 AM
StatusCode           : Successful
StatusMessage        :

ActivityId           : ce719997-086d-4c73-a2f1-ab703EXAMPLE
AutoScalingGroupName : my-asg
Cause                : At 2015-11-20T22:57:53Z a user request created an
 AutoScalingGroup changing the desired capacity
                       from 0 to 1.  At 2015-11-20T22:57:58Z an instance was
 started in response to a difference betwe
                      en desired and actual capacity, increasing the capacity
 from 0 to 1.
Description          : Launching a new EC2 instance: i-93633f9b
Details              : {"Availability Zone":"us-west-2b","Subnet
 ID":"subnet-5264e837"}
EndTime              : 11/20/2015 2:58:32 PM
Progress             : 100
StartTime            : 11/20/2015 2:57:59 PM
StatusCode           : Successful
StatusMessage        :
```

Ejemplo 2: En este ejemplo se describe la actividad de escalado especificada.

```
Get-ASScalingActivity -ActivityId "063308ae-aa22-4a9b-94f4-9fae4EXAMPLE"
```

Ejemplo 3: En este ejemplo se describen las actividades de escalado de las últimas seis semanas para todos los grupos de escalado automático.

```
Get-ASScalingActivity
```

- Para obtener más información sobre la API, consulte DescribeScalingActivitiesla referencia de Herramientas de AWS para PowerShell cmdlets (V5).

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 AutoScalingWrapper:
    """Encapsulates Amazon EC2 Auto Scaling actions."""

    def __init__(self, autoscaling_client):
        """
        :param autoscaling_client: A Boto3 Amazon EC2 Auto Scaling client.
        """
        self.autoscaling_client = autoscaling_client


    def describe_scaling_activities(self, group_name: str) -> List[Dict[str,
 Any]]:
        """
        Gets information about scaling activities for the group. Scaling
 activities
        are things like instances stopping or starting in response to user
 requests
        or capacity changes.

        :param group_name: The name of the group to look up.
        :return: A list of dictionaries representing the scaling activities for
 the
                 group, ordered with the most recent activity first.
        :raises ClientError: If there is an error describing the scaling
 activities.
        """
        try:
            paginator = self.autoscaling_client.get_paginator(
                "describe_scaling_activities"
            )
            response_iterator =
 paginator.paginate(AutoScalingGroupName=group_name)
```

```
            activities = []
            for response in response_iterator:
                activities.extend(response.get("Activities", []))

            logger.info(
                f"Successfully described scaling activities for group
'{group_name}'."
            )

        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            logger.error(
                f"Couldn't describe scaling activities for group '{group_name}'.
Error code: {error_code}, Message: {err.response['Error']['Message']}"
            )

            if error_code == "ResourceContentionFault":
                logger.error(
                    f"There is a conflict with another operation that is
modifying the Auto Scaling group '{group_name}'. "
                    "Please try again later."
                )
            raise
        else:
            return activities
```

- Para obtener más información sobre la API, consulta DescribeScalingActivitiesla 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.

```
    pub async fn describe_scenario(&self) -> AutoScalingScenarioDescription {
        let group = self
            .autoscaling
            .describe_auto_scaling_groups()
            .auto_scaling_group_names(self.auto_scaling_group_name.clone())
            .send()
            .await
            .map(|s| {
                s.auto_scaling_groups()
                    .iter()
                    .map(|s| {
                        format!(
                            "{}: {}",
                            s.auto_scaling_group_name().unwrap_or("Unknown"),
                            s.status().unwrap_or("Unknown")
                        )
                    })
                    .collect::<Vec<String>>()
            })
            .map_err(|e| {
                ScenarioError::new("Failed to describe auto scaling groups for
scenario", &e)
            });

        let instances = self
            .list_instances()
            .await
            .map_err(|e| anyhow!("There was an error listing instances: {e}",));

        // 10. DescribeScalingActivities: list the scaling activities that have
occurred for the group so far.
        //   Bonus: use CloudWatch API to get and show some metrics collected for
the group.
        //   CW.ListMetrics with Namespace='AWS/AutoScaling' and
Dimensions=[{'Name': 'AutoScalingGroupName', 'Value': }]
        //   CW.GetMetricStatistics with Statistics='Sum'. Start and End times
must be in UTC!
        let activities = self
            .autoscaling
            .describe_scaling_activities()
            .auto_scaling_group_name(self.auto_scaling_group_name.clone())
            .into_paginator()
            .items()
```

```
            .send()
            .collect::<Result<Vec<_>, _>>()
            .await
            .map_err(|e| {
                anyhow!(
                    "There was an error retrieving scaling activities: {}",
                    DisplayErrorContext(&e)
                )
            });

        AutoScalingScenarioDescription {
            group,
            instances,
            activities,
        }
    }
```

- Para obtener más información sobre la API, consulta DescribeScalingActivitiesla referencia sobre la API de AWS SDK para Rust.

SAP ABAP

SDK para 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.

```
" Example: iv_group_name = 'my-auto-scaling-group'

TRY.
    DATA(lo_output) = ao_asc->describescalingactivities(
      iv_autoscalinggroupname = iv_group_name ).

    ot_output = lo_output->get_activities( ).

    MESSAGE 'Scaling activities retrieved successfully' TYPE 'I'.
```

```
      CATCH /aws1/cx_ascresrccontionfault INTO DATA(lo_contention).
        RAISE EXCEPTION lo_contention.
      CATCH /aws1/cx_rt_generic INTO DATA(lo_generic_exception).
        RAISE EXCEPTION lo_generic_exception.
    ENDTRY.
```

- Para obtener más información sobre la API, consulte DescribeScalingActivitiesla 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, consulteUtilización de este servicio con un AWS SDK. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Utilizar **`DescribeScalingProcessTypes`** con una CLI

Los siguientes ejemplos de código muestran cómo utilizar `DescribeScalingProcessTypes`.

CLI

AWS CLI

Cómo describir los tipos de procesos disponibles

En este ejemplo se describen los tipos de procesos disponibles.

```
aws autoscaling describe-scaling-process-types
```

Salida:

```
{
    "Processes": [
        {
            "ProcessName": "AZRebalance"
        },
        {
            "ProcessName": "AddToLoadBalancer"
        },
        {
            "ProcessName": "AlarmNotification"
        },
```

```
        {
            "ProcessName": "HealthCheck"
        },
        {
            "ProcessName": "InstanceRefresh"
        },
        {
            "ProcessName": "Launch"
        },
        {
            "ProcessName": "ReplaceUnhealthy"
        },
        {
            "ProcessName": "ScheduledActions"
        },
        {
            "ProcessName": "Terminate"
        }
    ]
}
```

Para obtener más información, consulte Suspending and resuming scaling processes en la Guía del usuario de Amazon EC2 Auto Scaling.

- Para obtener más información sobre la API, consulte DescribeScalingProcessTypesla Referencia de AWS CLI comandos.

PowerShell

Herramientas para la PowerShell versión 4

Ejemplo 1: En este ejemplo se enumeran los tipos de procesos que admite el escalado automático.

```
Get-ASScalingProcessType
```

Salida:

```
ProcessName
-----------
AZRebalance
AddToLoadBalancer
```

```
AlarmNotification
HealthCheck
Launch
ReplaceUnhealthy
ScheduledActions
Terminate
```

- Para obtener más información sobre la API, consulte DescribeScalingProcessTypes Herramientas de AWS para PowerShellCmdlet Reference (V4).

Herramientas para la versión 5 PowerShell

Ejemplo 1: En este ejemplo se enumeran los tipos de procesos que admite el escalado automático.

```
Get-ASScalingProcessType
```

Salida:

```
ProcessName
-----------
AZRebalance
AddToLoadBalancer
AlarmNotification
HealthCheck
Launch
ReplaceUnhealthy
ScheduledActions
Terminate
```

- Para obtener más información sobre la API, consulte DescribeScalingProcessTypesla referencia de Herramientas de AWS para PowerShell cmdlets (V5).

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. Utilización de este servicio con un AWS SDK En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Utilizar **`DescribeScheduledActions`** con una CLI

Los siguientes ejemplos de código muestran cómo utilizar `DescribeScheduledActions`.

CLI

AWS CLI

Ejemplo 1: Cómo describir todas las acciones programadas

En este ejemplo, se describen todas las acciones programadas.

```
aws autoscaling describe-scheduled-actions
```

Salida:

```
{
    "ScheduledUpdateGroupActions": [
        {
            "AutoScalingGroupName": "my-asg",
            "ScheduledActionName": "my-recurring-action",
            "Recurrence": "30 0 1 1,6,12 *",
            "ScheduledActionARN": "arn:aws:autoscaling:us-
west-2:123456789012:scheduledUpdateGroupAction:8e86b655-b2e6-4410-8f29-
b4f094d6871c:autoScalingGroupName/my-asg:scheduledActionName/my-recurring-
action",
            "StartTime": "2023-12-01T04:00:00Z",
            "Time": "2023-12-01T04:00:00Z",
            "MinSize": 1,
            "MaxSize": 6,
            "DesiredCapacity": 4,
            "TimeZone": "America/New_York"
        }
    ]
}
```

Para obtener más información, consulte la sección sobre Escalado programado en la Guía del usuario de Amazon EC2 Auto Scaling.

Ejemplo 2: describir las acciones programadas para el grupo especificado

Para describir las acciones programadas para un grupo de escalado automático específico, utilice la opción `--auto-scaling-group-name`.

```
aws autoscaling describe-scheduled-actions \
```

```
    --auto-scaling-group-name my-asg
```

Salida:

```
{
    "ScheduledUpdateGroupActions": [
        {
            "AutoScalingGroupName": "my-asg",
            "ScheduledActionName": "my-recurring-action",
            "Recurrence": "30 0 1 1,6,12 *",
            "ScheduledActionARN": "arn:aws:autoscaling:us-
west-2:123456789012:scheduledUpdateGroupAction:8e86b655-b2e6-4410-8f29-
b4f094d6871c:autoScalingGroupName/my-asg:scheduledActionName/my-recurring-
action",
            "StartTime": "2023-12-01T04:00:00Z",
            "Time": "2023-12-01T04:00:00Z",
            "MinSize": 1,
            "MaxSize": 6,
            "DesiredCapacity": 4,
            "TimeZone": "America/New_York"
        }
    ]
}
```

Para obtener más información, consulte la sección sobre Escalado programado en la Guía del usuario de Amazon EC2 Auto Scaling.

Ejemplo 3: cómo describir la acción programada especificada

Para describir una acción programada específica, utilice la opción `--scheduled-action-names`.

```
aws autoscaling describe-scheduled-actions \
    --scheduled-action-names my-recurring-action
```

Salida:

```
{
    "ScheduledUpdateGroupActions": [
        {
            "AutoScalingGroupName": "my-asg",
            "ScheduledActionName": "my-recurring-action",
```

```
            "Recurrence": "30 0 1 1,6,12 *",
            "ScheduledActionARN": "arn:aws:autoscaling:us-
west-2:123456789012:scheduledUpdateGroupAction:8e86b655-b2e6-4410-8f29-
b4f094d6871c:autoScalingGroupName/my-asg:scheduledActionName/my-recurring-
action",
            "StartTime": "2023-12-01T04:00:00Z",
            "Time": "2023-12-01T04:00:00Z",
            "MinSize": 1,
            "MaxSize": 6,
            "DesiredCapacity": 4,
            "TimeZone": "America/New_York"
        }
    ]
}
```

Para obtener más información, consulte la sección sobre Escalado programado en la Guía del usuario de Amazon EC2 Auto Scaling.

Ejemplo 4: cómo describir las acciones programadas con una hora de inicio especifica

Para describir las acciones programadas que comienzan a una hora específica, utilice la opción `--start-time`.

```
aws autoscaling describe-scheduled-actions \
    --start-time "2023-12-01T04:00:00Z"
```

Salida:

```
{
    "ScheduledUpdateGroupActions": [
        {
            "AutoScalingGroupName": "my-asg",
            "ScheduledActionName": "my-recurring-action",
            "Recurrence": "30 0 1 1,6,12 *",
            "ScheduledActionARN": "arn:aws:autoscaling:us-
west-2:123456789012:scheduledUpdateGroupAction:8e86b655-b2e6-4410-8f29-
b4f094d6871c:autoScalingGroupName/my-asg:scheduledActionName/my-recurring-
action",
            "StartTime": "2023-12-01T04:00:00Z",
            "Time": "2023-12-01T04:00:00Z",
            "MinSize": 1,
            "MaxSize": 6,
            "DesiredCapacity": 4,
```

```
            "TimeZone": "America/New_York"
        }
    ]
}
```

Para obtener más información, consulte la sección sobre Escalado programado en la Guía del usuario de Amazon EC2 Auto Scaling.

Ejemplo 5: cómo describir las acciones programadas que finalizan a una hora específica

Para describir las acciones programadas que finalizan a una hora específica, utilice la opción `--end-time`.

```
aws autoscaling describe-scheduled-actions \
    --end-time "2023-12-01T04:00:00Z"
```

Salida:

```
{
    "ScheduledUpdateGroupActions": [
        {
            "AutoScalingGroupName": "my-asg",
            "ScheduledActionName": "my-recurring-action",
            "Recurrence": "30 0 1 1,6,12 *",
            "ScheduledActionARN": "arn:aws:autoscaling:us-
west-2:123456789012:scheduledUpdateGroupAction:8e86b655-b2e6-4410-8f29-
b4f094d6871c:autoScalingGroupName/my-asg:scheduledActionName/my-recurring-
action",
            "StartTime": "2023-12-01T04:00:00Z",
            "Time": "2023-12-01T04:00:00Z",
            "MinSize": 1,
            "MaxSize": 6,
            "DesiredCapacity": 4,
            "TimeZone": "America/New_York"
        }
    ]
}
```

Para obtener más información, consulte la sección sobre Escalado programado en la Guía del usuario de Amazon EC2 Auto Scaling.

Ejemplo 6: cómo describir un número específico de acciones programadas

Para devolver un número específico de acciones programadas, utilice la opción `--max-items`.

```
aws autoscaling describe-scheduled-actions \
    --auto-scaling-group-name my-asg \
    --max-items 1
```

Salida:

```
{
    "ScheduledUpdateGroupActions": [
        {
            "AutoScalingGroupName": "my-asg",
            "ScheduledActionName": "my-recurring-action",
            "Recurrence": "30 0 1 1,6,12 *",
            "ScheduledActionARN": "arn:aws:autoscaling:us-
west-2:123456789012:scheduledUpdateGroupAction:8e86b655-b2e6-4410-8f29-
b4f094d6871c:autoScalingGroupName/my-asg:scheduledActionName/my-recurring-
action",
            "StartTime": "2023-12-01T04:00:00Z",
            "Time": "2023-12-01T04:00:00Z",
            "MinSize": 1,
            "MaxSize": 6,
            "DesiredCapacity": 4,
            "TimeZone": "America/New_York"
        }
    ]
}
```

Si la salida incluye un campo de `NextToken`, hay más acciones programadas. Para obtener las acciones programadas adicionales, utilice el valor de este campo con la opción `--starting-token` en una llamada posterior de la siguiente manera.

```
aws autoscaling describe-scheduled-actions \
    --auto-scaling-group-name my-asg \
    --starting-token Z3M3LMPEXAMPLE
```

Para obtener más información, consulte la sección sobre Escalado programado en la Guía del usuario de Amazon EC2 Auto Scaling.

- Para obtener más información sobre la API, consulte DescribeScheduledActionsla Referencia de AWS CLI comandos.

PowerShell

Herramientas para la PowerShell versión 4

Ejemplo 1: En este ejemplo, se describen las acciones de escalado programadas para el grupo de escalado automático especificado.

```
Get-ASScheduledAction -AutoScalingGroupName my-asg
```

Salida:

```
AutoScalingGroupName : my-asg
DesiredCapacity      : 10
EndTime              :
MaxSize              :
MinSize              :
Recurrence           :
ScheduledActionARN   : arn:aws:autoscaling:us-
west-2:123456789012:scheduledUpdateGroupAction:8a4c5f24-6ec6-4306-a2dd-f7
                       2c3af3a4d6:autoScalingGroupName/my-
asg:scheduledActionName/myScheduledAction
ScheduledActionName  : myScheduledAction
StartTime            : 11/30/2015 8:00:00 AM
Time                 : 11/30/2015 8:00:00 AM
```

Ejemplo 2: En este ejemplo, se describen las acciones de escalado programadas especificadas.

```
Get-ASScheduledAction -ScheduledActionName @("myScheduledScaleOut",
 "myScheduledScaleIn")
```

Ejemplo 3: En este ejemplo, se describen las acciones de escalado programadas que comienzan a la hora especificada.

```
Get-ASScheduledAction -StartTime "2015-12-01T08:00:00Z"
```

Ejemplo 4: En este ejemplo, se describen las acciones de escalado programadas que finalizan a la hora especificada.

```
Get-ASScheduledAction -EndTime "2015-12-30T08:00:00Z"
```

Ejemplo 5: En este ejemplo, se describen las acciones de escalado programadas para todos los grupos de escalado automático.

```
Get-ASScheduledAction
```

- Para obtener más información sobre la API, consulte DescribeScheduledActions Herramientas de AWS para PowerShellCmdlet Reference (V4).

Herramientas para la versión 5 PowerShell

Ejemplo 1: En este ejemplo, se describen las acciones de escalado programadas para el grupo de escalado automático especificado.

```
Get-ASScheduledAction -AutoScalingGroupName my-asg
```

Salida:

```
AutoScalingGroupName : my-asg
DesiredCapacity      : 10
EndTime              :
MaxSize              :
MinSize              :
Recurrence           :
ScheduledActionARN   : arn:aws:autoscaling:us-
west-2:123456789012:scheduledUpdateGroupAction:8a4c5f24-6ec6-4306-a2dd-f7
                       2c3af3a4d6:autoScalingGroupName/my-
asg:scheduledActionName/myScheduledAction
ScheduledActionName  : myScheduledAction
StartTime            : 11/30/2015 8:00:00 AM
Time                 : 11/30/2015 8:00:00 AM
```

Ejemplo 2: En este ejemplo, se describen las acciones de escalado programadas especificadas.

```
Get-ASScheduledAction -ScheduledActionName @("myScheduledScaleOut",
 "myScheduledScaleIn")
```

Ejemplo 3: En este ejemplo, se describen las acciones de escalado programadas que comienzan a la hora especificada.

```
Get-ASScheduledAction -StartTime "2015-12-01T08:00:00Z"
```

Ejemplo 4: En este ejemplo, se describen las acciones de escalado programadas que finalizan a la hora especificada.

```
Get-ASScheduledAction -EndTime "2015-12-30T08:00:00Z"
```

Ejemplo 5: En este ejemplo, se describen las acciones de escalado programadas para todos los grupos de escalado automático.

```
Get-ASScheduledAction
```

- Para obtener más información sobre la API, consulte DescribeScheduledActionsla referencia de Herramientas de AWS para PowerShell cmdlets (V5).

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. Utilización de este servicio con un AWS SDK En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Utilizar **`DescribeTags`** con una CLI

Los siguientes ejemplos de código muestran cómo utilizar `DescribeTags`.

CLI

AWS CLI

Cómo describir todas las etiquetas

En este ejemplo, se describen todas las etiquetas.

```
aws autoscaling describe-tags
```

Salida:

```
{
    "Tags": [
        {
            "ResourceType": "auto-scaling-group",
            "ResourceId": "my-asg",
            "PropagateAtLaunch": true,
```

```
            "Value": "Research",
            "Key": "Dept"
        },
        {
            "ResourceType": "auto-scaling-group",
            "ResourceId": "my-asg",
            "PropagateAtLaunch": true,
            "Value": "WebServer",
            "Key": "Role"
        }
    ]
}
```

Para obtener más información, consulte Etiquetado de grupos e instancias de Auto Scaling en la Guía del usuario de Amazon EC2 Auto Scaling.

Ejemplo 2: describir las etiquetas de un grupo específico

Para describir etiquetas de un grupo de escalado automático específico, utilice la opción `--filters`.

```
aws autoscaling describe-tags --filters Name=auto-scaling-group,Values=my-asg
```

Para obtener más información, consulte Etiquetado de grupos e instancias de Auto Scaling en la Guía del usuario de Amazon EC2 Auto Scaling.

Ejemplo 3: cómo describir el número especificado de etiquetas

Para devolver un número específico de etiquetas, utilice la opción `--max-items`.

```
aws autoscaling describe-tags \
    --max-items 1
```

Si la salida incluye un campo de `NextToken`, hay más etiquetas. Para obtener las etiquetas adicionales, utilice el valor de este campo con la opción `--starting-token` en una llamada posterior de la siguiente manera.

```
aws autoscaling describe-tags \
    --filters Name=auto-scaling-group,Values=my-asg \
    --starting-token Z3M3LMPEXAMPLE
```

Para obtener más información, consulte Etiquetado de grupos e instancias de Auto Scaling en la Guía del usuario de Amazon EC2 Auto Scaling.

- Para obtener más información sobre la API, consulte DescribeTagsla Referencia de AWS CLI comandos.

PowerShell

Herramientas para la PowerShell versión 4

Ejemplo 1: En este ejemplo se describen las etiquetas con un valor clave de «myTag» o «myTag2». Los valores posibles para el nombre del filtro son auto-scaling-group '', 'clave', 'valor' y 'propagate-at-launch'. La sintaxis utilizada en este ejemplo requiere PowerShell la versión 3 o posterior.

```
Get-ASTag -Filter @( @{ Name="key"; Values=@("myTag", "myTag2") } )
```

Salida:

```
Key               : myTag2
PropagateAtLaunch : True
ResourceId        : my-asg
ResourceType      : auto-scaling-group
Value             : myTagValue2

Key               : myTag
PropagateAtLaunch : True
ResourceId        : my-asg
ResourceType      : auto-scaling-group
Value             : myTagValue
```

Ejemplo 2: Con la PowerShell versión 2, debe usar New-Object para crear el filtro para el parámetro Filter.

```
$keys = New-Object string[] 2
$keys[0] = "myTag"
$keys[1] = "myTag2"
$filter = New-Object Amazon.AutoScaling.Model.Filter
$filter.Name = "key"
$filter.Values = $keys
Get-ASTag -Filter @( $filter )
```

Ejemplo 3: En este ejemplo se describen todas las etiquetas para todos los grupos de escalado automático.

```
Get-ASTag
```

- Para obtener más información sobre la API, consulte DescribeTags Herramientas de AWS para PowerShellCmdlet Reference (V4).

Herramientas para la versión 5 PowerShell

Ejemplo 1: En este ejemplo se describen las etiquetas con un valor clave de «myTag» o «myTag2». Los valores posibles para el nombre del filtro son auto-scaling-group '', 'clave', 'valor' y 'propagate-at-launch'. La sintaxis utilizada en este ejemplo requiere PowerShell la versión 3 o posterior.

```
Get-ASTag -Filter @( @{ Name="key"; Values=@("myTag", "myTag2") } )
```

Salida:

```
Key               : myTag2
PropagateAtLaunch : True
ResourceId        : my-asg
ResourceType      : auto-scaling-group
Value             : myTagValue2

Key               : myTag
PropagateAtLaunch : True
ResourceId        : my-asg
ResourceType      : auto-scaling-group
Value             : myTagValue
```

Ejemplo 2: Con la PowerShell versión 2, debe usar New-Object para crear el filtro para el parámetro Filter.

```
$keys = New-Object string[] 2
$keys[0] = "myTag"
$keys[1] = "myTag2"
$filter = New-Object Amazon.AutoScaling.Model.Filter
$filter.Name = "key"
$filter.Values = $keys
```

```
Get-ASTag -Filter @( $filter )
```

Ejemplo 3: En este ejemplo se describen todas las etiquetas para todos los grupos de escalado automático.

```
Get-ASTag
```

- Para obtener más información sobre la API, consulte DescribeTags Herramientas de AWS para PowerShellCmdlet Reference (V5).

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. Utilización de este servicio con un AWS SDK En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Utilizar **`DescribeTerminationPolicyTypes`** con una CLI

Los siguientes ejemplos de código muestran cómo utilizar `DescribeTerminationPolicyTypes`.

CLI

AWS CLI

Cómo describir los tipos de políticas de terminación disponibles

En este ejemplo se describen los tipos de políticas de terminación disponibles.

```
aws autoscaling describe-termination-policy-types
```

Salida:

```
{
    "TerminationPolicyTypes": [
        "AllocationStrategy",
        "ClosestToNextInstanceHour",
        "Default",
        "NewestInstance",
        "OldestInstance",
        "OldestLaunchConfiguration",
        "OldestLaunchTemplate"
    ]
```

```
}
```

Para obtener más información, consulte el tema Controlar las instancias que Auto Scaling termina durante el escalado descendente en la guía del usuario de Amazon EC2 Auto Scaling.

- Para obtener más información sobre la API, consulte DescribeTerminationPolicyTypesla Referencia de AWS CLI comandos.

PowerShell

Herramientas para la PowerShell versión 4

Ejemplo 1: En este ejemplo se enumeran las políticas de terminación compatibles con el escalado automático.

```
Get-ASTerminationPolicyType
```

Salida:

```
ClosestToNextInstanceHour
Default
NewestInstance
OldestInstance
OldestLaunchConfiguration
```

- Para obtener más información sobre la API, consulte DescribeTerminationPolicyTypes Herramientas de AWS para PowerShellCmdlet Reference (V4).

Herramientas para la versión 5 PowerShell

Ejemplo 1: En este ejemplo se enumeran las políticas de terminación compatibles con el escalado automático.

```
Get-ASTerminationPolicyType
```

Salida:

```
ClosestToNextInstanceHour
Default
NewestInstance
OldestInstance
```

```
OldestLaunchConfiguration
```

- Para obtener más información sobre la API, consulte DescribeTerminationPolicyTypesla referencia de Herramientas de AWS para PowerShell cmdlets (V5).

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. Utilización de este servicio con un AWS SDK En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Utilizar **`DetachInstances`** con una CLI

Los siguientes ejemplos de código muestran cómo utilizar `DetachInstances`.

CLI

AWS CLI

Cómo desasociar una instancia de un grupo de escalado automático

En este ejemplo, se asocia la instancia especificada del grupo de escalado automático especificado.

```
aws autoscaling detach-instances \
    --instance-ids i-030017cfa84b20135 \
    --auto-scaling-group-name my-asg \
    --should-decrement-desired-capacity
```

Salida:

```
{
    "Activities": [
        {
            "ActivityId": "5091cb52-547a-47ce-a236-c9ccbc2cb2c9",
            "AutoScalingGroupName": "my-asg",
            "Description": "Detaching EC2 instance: i-030017cfa84b20135",
            "Cause": "At 2020-10-31T17:35:04Z instance i-030017cfa84b20135 was
 detached in response to a user request, shrinking the capacity from 2 to 1.",
            "StartTime": "2020-04-12T15:02:16.179Z",
            "StatusCode": "InProgress",
            "Progress": 50,
            "Details": "{\"Subnet ID\":\"subnet-6194ea3b\",\"Availability Zone\":
\"us-west-2c\"}"
```

```
        }
    ]
}
```

- Para obtener más información sobre la API, consulte DetachInstancesla Referencia de AWS CLI comandos.

PowerShell

Herramientas para la PowerShell versión 4

Ejemplo 1: En este ejemplo, se desasocia la instancia especificada del grupo de escalado automático especificado y se reduce la capacidad deseada para que Auto Scaling no inicialice una instancia de sustitución.

```
Dismount-ASInstance -InstanceId i-93633f9b -AutoScalingGroupName my-asg -
ShouldDecrementDesiredCapacity $true
```

Salida:

```
ActivityId           : 06733445-ce94-4039-be1b-b9f1866e276e
AutoScalingGroupName : my-asg
Cause                : At 2015-11-20T22:34:59Z instance i-93633f9b was detached
 in response to a user request, shrinking
                       the capacity from 2 to 1.
Description          : Detaching EC2 instance: i-93633f9b
Details              : {"Availability Zone":"us-west-2b","Subnet
 ID":"subnet-5264e837"}
EndTime              :
Progress             : 50
StartTime            : 11/20/2015 2:34:59 PM
StatusCode           : InProgress
StatusMessage        :
```

Ejemplo 2: En este ejemplo, se separa la instancia especificada del grupo de escalado automático especificado sin reducir la capacidad deseada. Auto Scaling inicializa una instancia de sustitución.

```
Dismount-ASInstance -InstanceId i-7bf746a2 -AutoScalingGroupName my-asg -
ShouldDecrementDesiredCapacity $false
```

Salida:

```
ActivityId           : f43a3cd4-d38c-4af7-9fe0-d76ec2307b6d
AutoScalingGroupName : my-asg
Cause                : At 2015-11-20T22:34:59Z instance i-7bf746a2 was detached
 in response to a user request.
Description          : Detaching EC2 instance: i-7bf746a2
Details              : {"Availability Zone":"us-west-2b","Subnet
 ID":"subnet-5264e837"}
EndTime              :
Progress             : 50
StartTime            : 11/20/2015 2:34:59 PM
StatusCode           : InProgress
StatusMessage        :
```

- Para obtener más información sobre la API, consulte DetachInstances Herramientas de AWS para PowerShellCmdlet Reference (V4).

Herramientas para la versión 5 PowerShell

Ejemplo 1: En este ejemplo, se desasocia la instancia especificada del grupo de escalado automático especificado y se reduce la capacidad deseada para que Auto Scaling no inicialice una instancia de sustitución.

```
Dismount-ASInstance -InstanceId i-93633f9b -AutoScalingGroupName my-asg -
ShouldDecrementDesiredCapacity $true
```

Salida:

```
ActivityId           : 06733445-ce94-4039-be1b-b9f1866e276e
AutoScalingGroupName : my-asg
Cause                : At 2015-11-20T22:34:59Z instance i-93633f9b was detached
 in response to a user request, shrinking
                       the capacity from 2 to 1.
Description          : Detaching EC2 instance: i-93633f9b
Details              : {"Availability Zone":"us-west-2b","Subnet
 ID":"subnet-5264e837"}
EndTime              :
Progress             : 50
StartTime            : 11/20/2015 2:34:59 PM
StatusCode           : InProgress
StatusMessage        :
```

Ejemplo 2: En este ejemplo, se separa la instancia especificada del grupo de escalado automático especificado sin reducir la capacidad deseada. Auto Scaling inicializa una instancia de sustitución.

```
Dismount-ASInstance -InstanceId i-7bf746a2 -AutoScalingGroupName my-asg -
ShouldDecrementDesiredCapacity $false
```

Salida:

```
ActivityId           : f43a3cd4-d38c-4af7-9fe0-d76ec2307b6d
AutoScalingGroupName : my-asg
Cause                : At 2015-11-20T22:34:59Z instance i-7bf746a2 was detached
 in response to a user request.
Description          : Detaching EC2 instance: i-7bf746a2
Details              : {"Availability Zone":"us-west-2b","Subnet
 ID":"subnet-5264e837"}
EndTime              :
Progress             : 50
StartTime            : 11/20/2015 2:34:59 PM
StatusCode           : InProgress
StatusMessage        :
```

- Para obtener más información sobre la API, consulte DetachInstancesla referencia de Herramientas de AWS para PowerShell cmdlets (V5).

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. Utilización de este servicio con un AWS SDK En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Utilizar **`DetachLoadBalancers`** con una CLI

Los siguientes ejemplos de código muestran cómo utilizar `DetachLoadBalancers`.

CLI

AWS CLI

Cómo desasociar un equilibrador de cargo clásico de un grupo de escalado automático

En este ejemplo, se asocia el equilibrador de carga clásico especificado del grupo de escalado automático especificado.

```
aws autoscaling detach-load-balancers \
    --load-balancer-names my-load-balancer \
    --auto-scaling-group-name my-asg
```

Este comando no genera ninguna salida.

Para obtener más información, consulte Adjuntar un equilibrador de carga al grupo de escalado automático en la Guía del usuario de Amazon EC2 Auto Scaling.

- Para obtener más información sobre la API, consulte DetachLoadBalancersla Referencia de AWS CLI comandos.

PowerShell

Herramientas para la PowerShell versión 4

Ejemplo 1: En este ejemplo se desasocia el equilibrador de carga especificado del grupo de escalado automático especificado.

```
Dismount-ASLoadBalancer -LoadBalancerName my-lb -AutoScalingGroupName my-asg
```

- Para obtener más información sobre la API, consulte DetachLoadBalancers Herramientas de AWS para PowerShellCmdlet Reference (V4).

Herramientas para la versión 5 PowerShell

Ejemplo 1: En este ejemplo se desasocia el equilibrador de carga especificado del grupo de escalado automático especificado.

```
Dismount-ASLoadBalancer -LoadBalancerName my-lb -AutoScalingGroupName my-asg
```

- Para obtener más información sobre la API, consulte DetachLoadBalancersla referencia de Herramientas de AWS para PowerShell cmdlets (V5).

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. Utilización de este servicio con un AWS SDK En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Úselo **`DisableMetricsCollection`** con un AWS SDK o CLI

Los siguientes ejemplos de código muestran cómo utilizar `DisableMetricsCollection`.

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:

- Conceptos básicos

.NET

SDK para .NET (v4)

> **Note**
>
> Hay más información GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
    /// <summary>
    /// Disable the collection of metric data for an Amazon EC2 Auto Scaling
    /// group.
    /// </summary>
    /// <param name="groupName">The name of the Auto Scaling group.</param>
    /// <returns>A Boolean value that indicates the success or failure of
    /// the operation.</returns>
    public async Task<bool> DisableMetricsCollectionAsync(string groupName)
    {
        var request = new DisableMetricsCollectionRequest
        {
            AutoScalingGroupName = groupName,
        };

        var response = await
_amazonAutoScaling.DisableMetricsCollectionAsync(request);
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

- Para obtener más información sobre la API, consulta DisableMetricsCollectionla Referencia AWS SDK para .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::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region (overrides config file).
        // clientConfig.region = "us-east-1";

    Aws::AutoScaling::AutoScalingClient autoScalingClient(clientConfig);

        Aws::AutoScaling::Model::DisableMetricsCollectionRequest request;
        request.SetAutoScalingGroupName(groupName);

        Aws::AutoScaling::Model::DisableMetricsCollectionOutcome outcome =
                autoScalingClient.DisableMetricsCollection(request);

        if (outcome.IsSuccess()) {
            std::cout << "Metrics collection has been disabled." << std::endl;
        }
        else {
            std::cerr << "Error with AutoScaling::DisableMetricsCollection. "
                      << outcome.GetError().GetMessage()
                      << std::endl;

        }
```

- Para obtener más información sobre la API, consulta DisableMetricsCollectionla Referencia AWS SDK para C++ de la API.

CLI

AWS CLI

Desactivación de la recopilación de métricas de grupo de un grupo de escalado automático

En este ejemplo, se deshabilita la recopilación de la métrica `GroupDesiredCapacity` del grupo de escalado automático especificado.

```
aws autoscaling disable-metrics-collection \
    --auto-scaling-group-name my-asg \
    --metrics GroupDesiredCapacity
```

Este comando no genera ninguna salida.

Para obtener más información, consulte Supervisión de CloudWatch las métricas de sus grupos e instancias de Auto Scaling en la Guía del usuario de Auto Scaling de Amazon EC2.

- Para obtener más información sobre la API, consulte DisableMetricsCollectionla 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 disableMetricsCollection(AutoScalingClient
autoScalingClient, String groupName) {
       try {
           DisableMetricsCollectionRequest disableMetricsCollectionRequest =
DisableMetricsCollectionRequest.builder()
                   .autoScalingGroupName(groupName)
                   .metrics("GroupMaxSize")
                   .build();


autoScalingClient.disableMetricsCollection(disableMetricsCollectionRequest);
           System.out.println("The disable metrics collection operation was
successful");

       } catch (AutoScalingException e) {
           System.err.println(e.awsErrorDetails().errorMessage());
```

```
            System.exit(1);
        }
    }
```

- Para obtener más información sobre la API, consulta DisableMetricsCollectionla Referencia AWS SDK for Java 2.x 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 disableMetricsCollection(groupName: String) {
    val disableMetricsCollectionRequest =
        DisableMetricsCollectionRequest {
            autoScalingGroupName = groupName
            metrics = listOf("GroupMaxSize")
        }

    AutoScalingClient { region = "us-east-1" }.use { autoScalingClient ->

 autoScalingClient.disableMetricsCollection(disableMetricsCollectionRequest)
        println("The disable metrics collection operation was successful")
    }
}
```

- Para obtener más información sobre la API, consulta DisableMetricsCollectionla referencia sobre el AWS SDK para la API de Kotlin.

PHP

SDK para PHP

> **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 function disableMetricsCollection($autoScalingGroupName)
    {
        return $this->autoScalingClient->disableMetricsCollection([
            'AutoScalingGroupName' => $autoScalingGroupName,
        ]);
    }
```

- Para obtener más información sobre la API, consulta DisableMetricsCollectionla Referencia AWS SDK para PHP de la API.

PowerShell

Herramientas para la PowerShell versión 4

Ejemplo 1: En este ejemplo se deshabilita la supervisión de las métricas especificadas para el grupo de escalado automático especificado.

```
Disable-ASMetricsCollection -AutoScalingGroupName my-asg -Metric
 @("GroupMinSize", "GroupMaxSize")
```

Ejemplo 2: En este ejemplo se deshabilita la supervisión de todas las métricas del grupo de escalado automático especificado.

```
Disable-ASMetricsCollection -AutoScalingGroupName my-asg
```

- Para obtener más información sobre la API, consulte DisableMetricsCollection Herramientas de AWS para PowerShellCmdlet Reference (V4).

Herramientas para la versión 5 PowerShell

Ejemplo 1: En este ejemplo se deshabilita la supervisión de las métricas especificadas para el grupo de escalado automático especificado.

```
Disable-ASMetricsCollection -AutoScalingGroupName my-asg -Metric
 @("GroupMinSize", "GroupMaxSize")
```

Ejemplo 2: En este ejemplo se deshabilita la supervisión de todas las métricas del grupo de escalado automático especificado.

```
Disable-ASMetricsCollection -AutoScalingGroupName my-asg
```

- Para obtener más información sobre la API, consulte DisableMetricsCollectionla referencia de Herramientas de AWS para PowerShell cmdlets (V5).

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 AutoScalingWrapper:
    """Encapsulates Amazon EC2 Auto Scaling actions."""

    def __init__(self, autoscaling_client):
        """
        :param autoscaling_client: A Boto3 Amazon EC2 Auto Scaling client.
        """
        self.autoscaling_client = autoscaling_client


    def disable_metrics(self, group_name: str) -> Dict[str, Any]:
        """
        Stops CloudWatch metric collection for the Auto Scaling group.
```

```
        :param group_name: The name of the group.
        :return: A dictionary with the response from disabling the metrics
collection.
        :raises ClientError: If there is an error disabling metrics collection.
        """
        try:
            response = self.autoscaling_client.disable_metrics_collection(
                AutoScalingGroupName=group_name
            )
            logger.info(
                f"Successfully disabled metrics collection for group
'{group_name}'."
            )
            return response
        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            logger.error(
                f"Couldn't disable metrics for group '{group_name}'. Error code:
{error_code}, Message: {err.response['Error']['Message']}"
            )

            if error_code == "ResourceContentionFault":
                logger.error(
                    f"There is a conflict with another operation that is
modifying the Auto Scaling group '{group_name}'. "
                    "Please try again later."
                )
            raise
```

- Para obtener más información sobre la API, consulta DisableMetricsCollectionla 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.

```
        // If this fails it's fine, just means there are extra cloudwatch metrics
events for the scale-down.
        let _ = self
            .autoscaling
            .disable_metrics_collection()
            .auto_scaling_group_name(self.auto_scaling_group_name.clone())
            .send()
            .await;
```

- Para obtener más información sobre la API, consulta DisableMetricsCollectionla referencia sobre la API de AWS SDK para Rust.

SAP ABAP

SDK para 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.

```
    " Example: iv_group_name = 'my-auto-scaling-group'

    TRY.
        ao_asc->disablemetricscollection(
          iv_autoscalinggroupname = iv_group_name ).

        MESSAGE 'Metrics collection disabled successfully' TYPE 'I'.
```

```
    CATCH /aws1/cx_ascresrccontionfault INTO DATA(lo_contention).
      RAISE EXCEPTION lo_contention.
    CATCH /aws1/cx_rt_generic INTO DATA(lo_generic_exception).
      RAISE EXCEPTION lo_generic_exception.
  ENDTRY.
```

- Para obtener más información sobre la API, consulte DisableMetricsCollectionla 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, consulteUtilización de este servicio con un AWS SDK. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Úselo **`EnableMetricsCollection`** con un AWS SDK o CLI

Los siguientes ejemplos de código muestran cómo utilizar `EnableMetricsCollection`.

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:

- Conceptos básicos

.NET

SDK para .NET (v4)

> **Note**
>
> Hay más información GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
/// <summary>
/// Enable the collection of metric data for an Auto Scaling group.
/// </summary>
/// <param name="groupName">The name of the Auto Scaling group.</param>
/// <returns>A Boolean value indicating the success of the action.</returns>
public async Task<bool> EnableMetricsCollectionAsync(string groupName)
```

```
    {
        var listMetrics = new List<string>
            {
                "GroupMaxSize",
            };

        var collectionRequest = new EnableMetricsCollectionRequest
        {
            AutoScalingGroupName = groupName,
            Metrics = listMetrics,
            Granularity = "1Minute",
        };

        var response = await
_amazonAutoScaling.EnableMetricsCollectionAsync(collectionRequest);
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

- Para obtener más información sobre la API, consulta EnableMetricsCollectionla Referencia AWS SDK para .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::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region (overrides config file).
        // clientConfig.region = "us-east-1";

    Aws::AutoScaling::AutoScalingClient autoScalingClient(clientConfig);

        Aws::AutoScaling::Model::EnableMetricsCollectionRequest request;
        request.SetAutoScalingGroupName(groupName);
```

```
        request.AddMetrics("GroupMinSize");
        request.AddMetrics("GroupMaxSize");
        request.AddMetrics("GroupDesiredCapacity");
        request.AddMetrics("GroupInServiceInstances");
        request.AddMetrics("GroupTotalInstances");
        request.SetGranularity("1Minute");

        Aws::AutoScaling::Model::EnableMetricsCollectionOutcome outcome =
                autoScalingClient.EnableMetricsCollection(request);
        if (outcome.IsSuccess()) {
            std::cout << "Auto Scaling metrics have been enabled."
                      << std::endl;
        }
        else {
            std::cerr << "Error with AutoScaling::EnableMetricsCollection. "
                      << outcome.GetError().GetMessage()
                      << std::endl;

        }
```

- Para obtener más información sobre la API, consulta EnableMetricsCollectionla Referencia AWS SDK para C++ de la API.

CLI

AWS CLI

Ejemplo 1: Habilitación de la recopilación de métricas de un grupo de escalado automático

En este ejemplo, se habilita la recopilación de datos para el grupo de escalado automático especificado.

```
aws autoscaling enable-metrics-collection \
    --auto-scaling-group-name my-asg \
    --granularity "1Minute"
```

Este comando no genera ninguna salida.

Para obtener más información, consulte Supervisión de CloudWatch las métricas de sus grupos e instancias de Auto Scaling en la Guía del usuario de Auto Scaling de Amazon EC2.

Ejemplo 2: Recopilación de datos para la métrica especificada para un grupo de escalado automático

Para recopilar datos para una métrica específica, utilice la opción `--metrics`.

```
aws autoscaling enable-metrics-collection \
    --auto-scaling-group-name my-asg \
    --metrics GroupDesiredCapacity --granularity "1Minute"
```

Este comando no genera ninguna salida.

Para obtener más información, consulte Supervisión de CloudWatch las métricas de sus grupos e instancias de Auto Scaling en la Guía del usuario de Auto Scaling de Amazon EC2.

- Para obtener más información sobre la API, consulte EnableMetricsCollectionla 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 enableMetricsCollection(AutoScalingClient
autoScalingClient, String groupName) {
       try {
           EnableMetricsCollectionRequest collectionRequest =
EnableMetricsCollectionRequest.builder()
                   .autoScalingGroupName(groupName)
                   .metrics("GroupMaxSize")
                   .granularity("1Minute")
                   .build();

           autoScalingClient.enableMetricsCollection(collectionRequest);
           System.out.println("The enable metrics collection operation was
successful");
```

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

- Para obtener más información sobre la API, consulta EnableMetricsCollectionla Referencia AWS SDK for Java 2.x 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 enableMetricsCollection(groupName: String?) {
    val collectionRequest =
        EnableMetricsCollectionRequest {
            autoScalingGroupName = groupName
            metrics = listOf("GroupMaxSize")
            granularity = "1Minute"
        }

    AutoScalingClient { region = "us-east-1" }.use { autoScalingClient ->
        autoScalingClient.enableMetricsCollection(collectionRequest)
        println("The enable metrics collection operation was successful")
    }
}
```

- Para obtener más información sobre la API, consulta EnableMetricsCollectionla referencia sobre el AWS SDK para la API de Kotlin.

PHP

SDK para PHP

> **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 function enableMetricsCollection($autoScalingGroupName, $granularity)
    {
        return $this->autoScalingClient->enableMetricsCollection([
            'AutoScalingGroupName' => $autoScalingGroupName,
            'Granularity' => $granularity,
        ]);
    }
```

- Para obtener más información sobre la API, consulta EnableMetricsCollectionla Referencia AWS SDK para PHP de la API.

PowerShell

Herramientas para la PowerShell versión 4

Ejemplo 1: En este ejemplo se habilita la supervisión de las métricas especificadas para el grupo de escalado automático especificado.

```
Enable-ASMetricsCollection  -Metric @("GroupMinSize", "GroupMaxSize") -
AutoScalingGroupName my-asg -Granularity 1Minute
```

Ejemplo 2: En este ejemplo se habilita la supervisión de todas las métricas del grupo de escalado automático especificado.

```
Enable-ASMetricsCollection -AutoScalingGroupName my-asg -Granularity 1Minute
```

- Para obtener más información sobre la API, consulte EnableMetricsCollection Herramientas de AWS para PowerShellCmdlet Reference (V4).

Herramientas para la versión 5 PowerShell

Ejemplo 1: En este ejemplo se habilita la supervisión de las métricas especificadas para el grupo de escalado automático especificado.

```
Enable-ASMetricsCollection  -Metric @("GroupMinSize", "GroupMaxSize") -
AutoScalingGroupName my-asg -Granularity 1Minute
```

Ejemplo 2: En este ejemplo se habilita la supervisión de todas las métricas del grupo de escalado automático especificado.

```
Enable-ASMetricsCollection -AutoScalingGroupName my-asg -Granularity 1Minute
```

- Para obtener más información sobre la API, consulte EnableMetricsCollectionla referencia de Herramientas de AWS para PowerShell cmdlets (V5).

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 AutoScalingWrapper:
    """Encapsulates Amazon EC2 Auto Scaling actions."""

    def __init__(self, autoscaling_client):
        """
        :param autoscaling_client: A Boto3 Amazon EC2 Auto Scaling client.
        """
        self.autoscaling_client = autoscaling_client


    def enable_metrics(self, group_name: str, metrics: List[str]) -> Dict[str,
 Any]:
        """
```

```
        Enables CloudWatch metric collection for Amazon EC2 Auto Scaling
activities.

        :param group_name: The name of the group to enable.
        :param metrics: A list of metrics to collect.
        :return: A dictionary with the response from enabling the metrics
collection.
        :raises ClientError: If there is an error enabling metrics collection.
        """
        try:
            response = self.autoscaling_client.enable_metrics_collection(
                AutoScalingGroupName=group_name, Metrics=metrics,
Granularity="1Minute"
            )
            logger.info(
                f"Successfully enabled metrics for Auto Scaling group
'{group_name}'."
            )

        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            logger.error(
                f"Couldn't enable metrics on '{group_name}'. Error code:
{error_code}, Message: {err.response['Error']['Message']}"
            )

            if error_code == "ResourceContentionFault":
                logger.error(
                    f"There is a conflict with another operation that is
modifying the Auto Scaling group '{group_name}'. "
                    "Please try again later."
                )
            elif error_code == "InvalidParameterCombination":
                logger.error(
                    f"The combination of parameters provided for enabling metrics
on '{group_name}' is not valid. "
                    "Please check the parameters and try again."
                )
            raise
        else:
            return response
```

- Para obtener más información sobre la API, consulta EnableMetricsCollectionla 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.

```
        let enable_metrics_collection = autoscaling
            .enable_metrics_collection()
            .auto_scaling_group_name(auto_scaling_group_name.as_str())
            .granularity("1Minute")
            .set_metrics(Some(vec![
                String::from("GroupMinSize"),
                String::from("GroupMaxSize"),
                String::from("GroupDesiredCapacity"),
                String::from("GroupInServiceInstances"),
                String::from("GroupTotalInstances"),
            ]))
            .send()
            .await;
```

- Para obtener más información sobre la API, consulta EnableMetricsCollectionla referencia sobre la API de AWS SDK para Rust.

SAP ABAP

SDK para 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.

```
    " Example: iv_group_name = 'my-auto-scaling-group'
    " Example: it_metrics contains list of metrics like 'GroupMinSize',
 'GroupMaxSize', etc.

    TRY.
        ao_asc->enablemetricscollection(
          iv_autoscalinggroupname = iv_group_name
          it_metrics = it_metrics
          iv_granularity = '1Minute' ).

        MESSAGE 'Metrics collection enabled successfully' TYPE 'I'.

      CATCH /aws1/cx_ascresrccontionfault INTO DATA(lo_contention).
        RAISE EXCEPTION lo_contention.
      CATCH /aws1/cx_rt_generic INTO DATA(lo_generic_exception).
        RAISE EXCEPTION lo_generic_exception.
    ENDTRY.
```

- Para obtener más información sobre la API, consulte EnableMetricsCollectionla 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, consulteUtilización de este servicio con un AWS SDK. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Utilizar **`EnterStandby`** con una CLI

Los siguientes ejemplos de código muestran cómo utilizar `EnterStandby`.

CLI

AWS CLI

Cómo mover las instancias al modo de espera

En este ejemplo, se pone la instancia especificada en modo de espera. Esto resulta útil para actualizar o solucionar problemas de una instancia que está actualmente en servicio.

```
aws autoscaling enter-standby \
    --instance-ids i-061c63c5eb45f0416 \
    --auto-scaling-group-name my-asg \
```

```
    --should-decrement-desired-capacity
```

Salida:

```
{
    "Activities": [
        {
            "ActivityId": "ffa056b4-6ed3-41ba-ae7c-249dfae6eba1",
            "AutoScalingGroupName": "my-asg",
            "Description": "Moving EC2 instance to Standby: i-061c63c5eb45f0416",
            "Cause": "At 2020-10-31T20:31:00Z instance i-061c63c5eb45f0416 was
 moved to standby in response to a user request, shrinking the capacity from 1 to
 0.",
            "StartTime": "2020-10-31T20:31:00.949Z",
            "StatusCode": "InProgress",
            "Progress": 50,
            "Details": "{\"Subnet ID\":\"subnet-6194ea3b\",\"Availability Zone\":
\"us-west-2c\"}"
        }
    ]
}
```

Para obtener más información, consulte Amazon EC2 Auto Scaling instance lifecycle en la Guía del usuario de Amazon EC2 Auto Scaling.

- Para obtener más información sobre la API, consulte EnterStandbyla Referencia de AWS CLI comandos.

PowerShell

Herramientas para la PowerShell versión 4

Ejemplo 1: En este ejemplo, se pone la instancia especificada en modo de espera y se reduce la capacidad deseada para que Auto Scaling no inicialice una instancia de sustitución.

```
Enter-ASStandby -InstanceId i-93633f9b -AutoScalingGroupName my-asg -
ShouldDecrementDesiredCapacity $true
```

Salida:

```
ActivityId           : e36a5a54-ced6-4df8-bd19-708e2a59a649
```

```
AutoScalingGroupName : my-asg
Cause                : At 2015-11-22T15:48:06Z instance i-95b8484f was moved to
 standby in response to a user request,
                       shrinking the capacity from 2 to 1.
Description          : Moving EC2 instance to Standby: i-95b8484f
Details              : {"Availability Zone":"us-west-2b","Subnet
 ID":"subnet-5264e837"}
EndTime              :
Progress             : 50
StartTime            : 11/22/2015 7:48:06 AM
StatusCode           : InProgress
StatusMessage        :
```

Ejemplo 2: En este ejemplo, se pone la instancia especificada en modo de espera sin reducir la capacidad deseada. Auto Scaling inicializa una instancia de sustitución.

```
Enter-ASStandby -InstanceId i-93633f9b -AutoScalingGroupName my-asg -
ShouldDecrementDesiredCapacity $false
```

Salida:

```
ActivityId           : e36a5a54-ced6-4df8-bd19-708e2a59a649
AutoScalingGroupName : my-asg
Cause                : At 2015-11-22T15:48:06Z instance i-95b8484f was moved to
 standby in response to a user request.
Description          : Moving EC2 instance to Standby: i-95b8484f
Details              : {"Availability Zone":"us-west-2b","Subnet
 ID":"subnet-5264e837"}
EndTime              :
Progress             : 50
StartTime            : 11/22/2015 7:48:06 AM
StatusCode           : InProgress
StatusMessage        :
```

- Para obtener más información sobre la API, consulte EnterStandby Herramientas de AWS para PowerShellCmdlet Reference (V4).

Herramientas para la versión 5 PowerShell

Ejemplo 1: En este ejemplo, se pone la instancia especificada en modo de espera y se reduce la capacidad deseada para que Auto Scaling no inicialice una instancia de sustitución.

```
Enter-ASStandby -InstanceId i-93633f9b -AutoScalingGroupName my-asg -
ShouldDecrementDesiredCapacity $true
```

Salida:

```
ActivityId           : e36a5a54-ced6-4df8-bd19-708e2a59a649
AutoScalingGroupName : my-asg
Cause                : At 2015-11-22T15:48:06Z instance i-95b8484f was moved to
 standby in response to a user request,
                       shrinking the capacity from 2 to 1.
Description          : Moving EC2 instance to Standby: i-95b8484f
Details              : {"Availability Zone":"us-west-2b","Subnet
 ID":"subnet-5264e837"}
EndTime              :
Progress             : 50
StartTime            : 11/22/2015 7:48:06 AM
StatusCode           : InProgress
StatusMessage        :
```

Ejemplo 2: En este ejemplo, se pone la instancia especificada en modo de espera sin reducir la capacidad deseada. Auto Scaling inicializa una instancia de sustitución.

```
Enter-ASStandby -InstanceId i-93633f9b -AutoScalingGroupName my-asg -
ShouldDecrementDesiredCapacity $false
```

Salida:

```
ActivityId           : e36a5a54-ced6-4df8-bd19-708e2a59a649
AutoScalingGroupName : my-asg
Cause                : At 2015-11-22T15:48:06Z instance i-95b8484f was moved to
 standby in response to a user request.
Description          : Moving EC2 instance to Standby: i-95b8484f
Details              : {"Availability Zone":"us-west-2b","Subnet
 ID":"subnet-5264e837"}
EndTime              :
Progress             : 50
StartTime            : 11/22/2015 7:48:06 AM
StatusCode           : InProgress
StatusMessage        :
```

- Para obtener más información sobre la API, consulte EnterStandbyla referencia de Herramientas de AWS para PowerShell cmdlets (V5).

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. Utilización de este servicio con un AWS SDK En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Utilizar **`ExecutePolicy`** con una CLI

Los siguientes ejemplos de código muestran cómo utilizar `ExecutePolicy`.

CLI

AWS CLI

Cómo ejecutar una política de escalado

En este ejemplo, se ejecuta la política de escalado denominada `my-step-scale-out-policy` para el grupo de escalado automático especificado.

```
aws autoscaling execute-policy \
    --auto-scaling-group-name my-asg \
    --policy-name  my-step-scale-out-policy \
    --metric-value 95 \
    --breach-threshold 80
```

Este comando no genera ninguna salida.

Para obtener más información, consulte Step and simple scaling policies en la Guía del usuario de Amazon EC2 Auto Scaling.

- Para obtener más información sobre la API, consulte ExecutePolicyla Referencia de AWS CLI comandos.

PowerShell

Herramientas para la PowerShell versión 4

Ejemplo 1: En este ejemplo se ejecuta la política especificada para el grupo de escalado automático especificado.

```
Start-ASPolicy -AutoScalingGroupName my-asg -PolicyName "myScaleInPolicy"
```

Ejemplo 2: En este ejemplo se ejecuta la política especificada para el grupo de escalado automático especificado, después de esperar a que finalice el período de recuperación.

```
Start-ASPolicy -AutoScalingGroupName my-asg -PolicyName "myScaleInPolicy" -
HonorCooldown $true
```

- Para obtener más información sobre la API, consulte ExecutePolicy Herramientas de AWS para PowerShellCmdlet Reference (V4).

Herramientas para la versión 5 PowerShell

Ejemplo 1: En este ejemplo se ejecuta la política especificada para el grupo de escalado automático especificado.

```
Start-ASPolicy -AutoScalingGroupName my-asg -PolicyName "myScaleInPolicy"
```

Ejemplo 2: En este ejemplo se ejecuta la política especificada para el grupo de escalado automático especificado, después de esperar a que finalice el período de recuperación.

```
Start-ASPolicy -AutoScalingGroupName my-asg -PolicyName "myScaleInPolicy" -
HonorCooldown $true
```

- Para obtener más información sobre la API, consulte ExecutePolicyla referencia de Herramientas de AWS para PowerShell cmdlets (V5).

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. Utilización de este servicio con un AWS SDK En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Utilizar **`ExitStandby`** con una CLI

Los siguientes ejemplos de código muestran cómo utilizar `ExitStandby`.

CLI

AWS CLI

Cómo sacar las instancias del modo de espera

En este ejemplo se sacan las instancias especificadas del modo de espera.

```
aws autoscaling exit-standby \
    --instance-ids i-061c63c5eb45f0416 \
    --auto-scaling-group-name my-asg
```

Salida:

```
{
    "Activities": [
        {
            "ActivityId": "142928e1-a2dc-453a-9b24-b85ad6735928",
            "AutoScalingGroupName": "my-asg",
            "Description": "Moving EC2 instance out of Standby:
 i-061c63c5eb45f0416",
            "Cause": "At 2020-10-31T20:32:50Z instance i-061c63c5eb45f0416 was
 moved out of standby in response to a user request, increasing the capacity from
 0 to 1.",
            "StartTime": "2020-10-31T20:32:50.222Z",
            "StatusCode": "PreInService",
            "Progress": 30,
            "Details": "{\"Subnet ID\":\"subnet-6194ea3b\",\"Availability Zone\":
\"us-west-2c\"}"
        }
    ]
}
```

Para obtener más información, consulte Eliminación temporal de instancias de su grupo de Auto Scaling en la Guía del usuario de Amazon EC2 Auto Scaling.

- Para obtener más información sobre la API, consulte ExitStandbyla Referencia de AWS CLI comandos.

PowerShell

Herramientas para la PowerShell versión 4

Ejemplo 1: En este ejemplo se saca la instancia especificada del modo de espera.

```
Exit-ASStandby -InstanceId i-93633f9b -AutoScalingGroupName my-asg
```

Salida:

```
ActivityId           : 1833d3e8-e32f-454e-b731-0670ad4c6934
AutoScalingGroupName : my-asg
Cause                : At 2015-11-22T15:51:21Z instance i-95b8484f was moved out
 of standby in response to a user
                       request, increasing the capacity from 1 to 2.
Description          : Moving EC2 instance out of Standby: i-95b8484f
Details              : {"Availability Zone":"us-west-2b","Subnet
 ID":"subnet-5264e837"}
EndTime              :
Progress             : 30
StartTime            : 11/22/2015 7:51:21 AM
StatusCode           : PreInService
StatusMessage        :
```

- Para obtener más información sobre la API, consulte ExitStandby Herramientas de AWS para PowerShellCmdlet Reference (V4).

Herramientas para la versión 5 PowerShell

Ejemplo 1: En este ejemplo se saca la instancia especificada del modo de espera.

```
Exit-ASStandby -InstanceId i-93633f9b -AutoScalingGroupName my-asg
```

Salida:

```
ActivityId           : 1833d3e8-e32f-454e-b731-0670ad4c6934
AutoScalingGroupName : my-asg
Cause                : At 2015-11-22T15:51:21Z instance i-95b8484f was moved out
 of standby in response to a user
                       request, increasing the capacity from 1 to 2.
Description          : Moving EC2 instance out of Standby: i-95b8484f
Details              : {"Availability Zone":"us-west-2b","Subnet
 ID":"subnet-5264e837"}
EndTime              :
Progress             : 30
StartTime            : 11/22/2015 7:51:21 AM
StatusCode           : PreInService
StatusMessage        :
```

- Para obtener más información sobre la API, consulte ExitStandbyla referencia de Herramientas de AWS para PowerShell cmdlets (V5).

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. Utilización de este servicio con un AWS SDK En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Utilizar **`PutLifecycleHook`** con una CLI

Los siguientes ejemplos de código muestran cómo utilizar `PutLifecycleHook`.

CLI

AWS CLI

Ejemplo 1: Cómo crear un enlace de ciclo de vida

En este ejemplo, se crea un enlace de ciclo de vida que se invocará en cualquier instancia recién inicializada, con un tiempo de espera de 4800 segundos. Esto resulta útil para mantener las instancias en estado de espera hasta que finalicen los scripts de datos de usuario o para invocar una función AWS Lambda mediante. EventBridge

```
aws autoscaling put-lifecycle-hook \
    --auto-scaling-group-name my-asg \
    --lifecycle-hook-name my-launch-hook \
    --lifecycle-transition autoscaling:EC2_INSTANCE_LAUNCHING \
    --heartbeat-timeout 4800
```

Este comando no genera ninguna salida. Si ya existe un enlace de ciclo de vida con el mismo nombre, el nuevo enlace de ciclo de vida lo sobrescribirá.

Para obtener más información, consulte Enlaces de ciclo de vida de Amazon EC2 Auto Scaling en la guía del usuario de Amazon EC2 Auto Scaling.

Ejemplo 2: enviar un mensaje de correo electrónico de Amazon SNS para notificarle las transiciones de estado de la instancia

En este ejemplo, se crea un enlace de ciclo de vida con el tema de Amazon SNS y rol de IAM que se debe utilizar para recibir notificaciones en la inicialización de la instancia.

```
aws autoscaling put-lifecycle-hook \
    --auto-scaling-group-name my-asg \
    --lifecycle-hook-name my-launch-hook \
    --lifecycle-transition autoscaling:EC2_INSTANCE_LAUNCHING \
    --notification-target-arn arn:aws:sns:us-west-2:123456789012:my-sns-topic \
```

```
    --role-arn arn:aws:iam::123456789012:role/my-auto-scaling-role
```

Este comando no genera ninguna salida.

Para obtener más información, consulte Enlaces de ciclo de vida de Amazon EC2 Auto Scaling en la guía del usuario de Amazon EC2 Auto Scaling.

Ejemplo 3: cómo publicar un mensaje en una cola de Amazon SQS

En este ejemplo, se crea un enlace de ciclo de vida que publica un mensaje con metadatos en la cola de Amazon SQS especificada.

```
aws autoscaling put-lifecycle-hook \
    --auto-scaling-group-name my-asg \
    --lifecycle-hook-name my-launch-hook \
    --lifecycle-transition autoscaling:EC2_INSTANCE_LAUNCHING \
    --notification-target-arn arn:aws:sqs:us-west-2:123456789012:my-sqs-queue \
    --role-arn arn:aws:iam::123456789012:role/my-notification-role \
    --notification-metadata "SQS message metadata"
```

Este comando no genera ninguna salida.

Para obtener más información, consulte Enlaces de ciclo de vida de Amazon EC2 Auto Scaling en la guía del usuario de Amazon EC2 Auto Scaling.

- Para obtener más información sobre la API, consulte la Referencia PutLifecycleHookde AWS CLI comandos.

PowerShell

Herramientas para la PowerShell versión 4

Ejemplo 1: En este ejemplo se agrega el enlace de ciclo de vida especificado al grupo de escalado automático especificado.

```
Write-ASLifecycleHook -AutoScalingGroupName my-asg -LifecycleHookName
 "myLifecycleHook" -LifecycleTransition "autoscaling:EC2_INSTANCE_LAUNCHING" -
NotificationTargetARN "arn:aws:sns:us-west-2:123456789012:my-sns-topic" -RoleARN
 "arn:aws:iam::123456789012:role/my-iam-role"
```

- Para obtener más información sobre la API, consulte PutLifecycleHook Herramientas de AWS para PowerShellCmdlet Reference (V4).

Herramientas para la versión 5 PowerShell

Ejemplo 1: En este ejemplo se agrega el enlace de ciclo de vida especificado al grupo de escalado automático especificado.

```
Write-ASLifecycleHook -AutoScalingGroupName my-asg -LifecycleHookName
 "myLifecycleHook" -LifecycleTransition "autoscaling:EC2_INSTANCE_LAUNCHING" -
NotificationTargetARN "arn:aws:sns:us-west-2:123456789012:my-sns-topic" -RoleARN
 "arn:aws:iam::123456789012:role/my-iam-role"
```

- Para obtener más información sobre la API, consulte PutLifecycleHookla referencia de Herramientas de AWS para PowerShell cmdlets (V5).

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. Utilización de este servicio con un AWS SDK En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Utilizar **PutNotificationConfiguration** con una CLI

Los siguientes ejemplos de código muestran cómo utilizar `PutNotificationConfiguration`.

CLI

AWS CLI

Cómo añadir una nueva notificación

En este ejemplo, se añade la notificación especificada al grupo de escalado automático especificado.

```
aws autoscaling put-notification-configuration \
    --auto-scaling-group-name my-asg \
    --topic-arn arn:aws:sns:us-west-2:123456789012:my-sns-topic \
    --notification-type autoscaling:TEST_NOTIFICATION
```

Este comando no genera ninguna salida.

Para obtener más información, consulte Obtener notificaciones de Amazon SNS cuando su grupo de Auto Scaling escala en la Guía del usuario de Amazon EC2 Auto Scaling.

- Para obtener más información sobre la API, consulte PutNotificationConfigurationla Referencia de AWS CLI comandos.

PowerShell

Herramientas para la PowerShell versión 4

Ejemplo 1: En este ejemplo se configura el grupo de escalado automático especificado para enviar una notificación al tema de SNS especificado cuando inicialice instancias de EC2.

```
Write-ASNotificationConfiguration -AutoScalingGroupName my-asg -
NotificationType "autoscaling:EC2_INSTANCE_LAUNCH" -TopicARN "arn:aws:sns:us-
west-2:123456789012:my-topic"
```

Ejemplo 2: En este ejemplo se configura el grupo de escalado automático especificado para enviar una notificación al tema de SNS especificado cuando inicialice o finalice las instancias de EC2.

```
Write-ASNotificationConfiguration -AutoScalingGroupName my-asg -NotificationType
 @("autoscaling:EC2_INSTANCE_LAUNCH", "autoscaling:EC2_INSTANCE_TERMINATE") -
TopicARN "arn:aws:sns:us-west-2:123456789012:my-topic"
```

- Para obtener más información sobre la API, consulte PutNotificationConfiguration Herramientas de AWS para PowerShellCmdlet Reference (V4).

Herramientas para la versión 5 PowerShell

Ejemplo 1: En este ejemplo se configura el grupo de escalado automático especificado para enviar una notificación al tema de SNS especificado cuando inicialice instancias de EC2.

```
Write-ASNotificationConfiguration -AutoScalingGroupName my-asg -
NotificationType "autoscaling:EC2_INSTANCE_LAUNCH" -TopicARN "arn:aws:sns:us-
west-2:123456789012:my-topic"
```

Ejemplo 2: En este ejemplo se configura el grupo de escalado automático especificado para enviar una notificación al tema de SNS especificado cuando inicialice o finalice las instancias de EC2.

```
Write-ASNotificationConfiguration -AutoScalingGroupName my-asg -NotificationType
 @("autoscaling:EC2_INSTANCE_LAUNCH", "autoscaling:EC2_INSTANCE_TERMINATE") -
TopicARN "arn:aws:sns:us-west-2:123456789012:my-topic"
```

- Para obtener más información sobre la API, consulte PutNotificationConfigurationla referencia de Herramientas de AWS para PowerShell cmdlets (V5).

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. Utilización de este servicio con un AWS SDK En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Utilizar **`PutScalingPolicy`** con una CLI

Los siguientes ejemplos de código muestran cómo utilizar `PutScalingPolicy`.

CLI

AWS CLI

Cómo añadir una política de escalado de seguimiento de destino a un grupo de escalado automático

El siguiente ejemplo de `put-scaling-policy` aplica una política de escalado de seguimiento de destino al grupo de escalado automático especificado. El resultado contiene los nombres ARNs y los nombres de las dos CloudWatch alarmas creadas en su nombre. Si existe una política de escalado con el mismo nombre, la nueva política la sobrescribirá.

```
aws autoscaling put-scaling-policy --auto-scaling-group-name my-asg \
  --policy-name alb1000-target-tracking-scaling-policy \
  --policy-type TargetTrackingScaling \
  --target-tracking-configuration file://config.json
```

Contenido de `config.json`:

```
{
     "TargetValue": 1000.0,
     "PredefinedMetricSpecification": {
          "PredefinedMetricType": "ALBRequestCountPerTarget",
          "ResourceLabel": "app/my-alb/778d41231b141a0f/targetgroup/my-alb-
target-group/943f017f100becff"
     }
}
```

Salida:

```
{
     "PolicyARN": "arn:aws:autoscaling:region:account-id:scalingPolicy:228f02c2-
c665-4bfd-aaac-8b04080bea3c:autoScalingGroupName/my-asg:policyName/alb1000-
target-tracking-scaling-policy",
```

```
    "Alarms": [
        {
            "AlarmARN": "arn:aws:cloudwatch:region:account-
id:alarm:TargetTracking-my-asg-AlarmHigh-fc0e4183-23ac-497e-9992-691c9980c38e",
            "AlarmName": "TargetTracking-my-asg-AlarmHigh-
fc0e4183-23ac-497e-9992-691c9980c38e"
        },
        {
            "AlarmARN": "arn:aws:cloudwatch:region:account-
id:alarm:TargetTracking-my-asg-AlarmLow-61a39305-ed0c-47af-bd9e-471a352ee1a2",
            "AlarmName": "TargetTracking-my-asg-AlarmLow-61a39305-ed0c-47af-
bd9e-471a352ee1a2"
        }
    ]
}
```

Para ver más ejemplos, consulte Ejemplos de políticas de escalado para la interfaz de línea de AWS comandos (AWS CLI) en la Guía del usuario de Amazon EC2 Auto Scaling.

- Para obtener más información sobre la API, consulte PutScalingPolicyla Referencia de AWS CLI comandos.

PowerShell

Herramientas para la PowerShell versión 4

Ejemplo 1: En este ejemplo se agrega la política especificada al grupo de escalado automático especificado. El tipo de ajuste especificado determina cómo interpretar el ScalingAdjustment parámetro. Con 'ChangeInCapacity', un valor positivo aumenta la capacidad en el número especificado de instancias y un valor negativo reduce la capacidad en el número especificado de instancias.

```
Write-ASScalingPolicy -AutoScalingGroupName my-asg -AdjustmentType
 "ChangeInCapacity" -PolicyName "myScaleInPolicy" -ScalingAdjustment -1
```

Salida:

```
arn:aws:autoscaling:us-west-2:123456789012:scalingPolicy:aa3836ab-5462-42c7-adab-
e1d769fc24ef:autoScalingGroupName/my-asg
:policyName/myScaleInPolicy
```

- Para obtener información sobre la API, consulte PutScalingPolicy Herramientas de AWS para PowerShellCmdlet Reference (V4).

Herramientas para la versión 5 PowerShell

Ejemplo 1: En este ejemplo se agrega la política especificada al grupo de escalado automático especificado. El tipo de ajuste especificado determina cómo interpretar el ScalingAdjustment parámetro. Con 'ChangeInCapacity', un valor positivo aumenta la capacidad en el número especificado de instancias y un valor negativo reduce la capacidad en el número especificado de instancias.

```
Write-ASScalingPolicy -AutoScalingGroupName my-asg -AdjustmentType
 "ChangeInCapacity" -PolicyName "myScaleInPolicy" -ScalingAdjustment -1
```

Salida:

```
arn:aws:autoscaling:us-west-2:123456789012:scalingPolicy:aa3836ab-5462-42c7-adab-
e1d769fc24ef:autoScalingGroupName/my-asg
:policyName/myScaleInPolicy
```

- Para obtener más información sobre la API, consulte PutScalingPolicy Herramientas de AWS para PowerShellCmdlet Reference (V5).

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. Utilización de este servicio con un AWS SDK En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Utilizar **PutScheduledUpdateGroupAction** con una CLI

Los siguientes ejemplos de código muestran cómo utilizar `PutScheduledUpdateGroupAction`.

CLI

AWS CLI

Ejemplo 1: Cómo añadir una acción programada a un grupo de escalado automático

En este ejemplo, se añade la acción programada especificada al grupo de escalado automático especificado.

```
aws autoscaling put-scheduled-update-group-action \
```

```
    --auto-scaling-group-name my-asg \
    --scheduled-action-name my-scheduled-action \
    --start-time "2023-05-12T08:00:00Z" \
    --min-size 2 \
    --max-size 6 \
    --desired-capacity 4
```

Este comando no genera ninguna salida. Si existe una acción programada con el mismo nombre, la nueva acción programada la sobrescribirá.

Para ver más ejemplos, consulte Escalado programado en la Guía del usuario de Amazon EC2 Auto Scaling.

Ejemplo 2: especificar una programación recurrente

En este ejemplo, se crea una acción programada para escalar un programa recurrente cuya ejecución está prevista para las 00:30 h del primero de enero, junio y diciembre de cada año.

```
aws autoscaling put-scheduled-update-group-action \
    --auto-scaling-group-name my-asg \
    --scheduled-action-name my-recurring-action \
    --recurrence "30 0 1 1,6,12 *" \
    --min-size 2 \
    --max-size 6 \
    --desired-capacity 4
```

Este comando no genera ninguna salida. Si existe una acción programada con el mismo nombre, la nueva acción programada la sobrescribirá.

Para obtener más información, consulte Escalado programado en la Guía del usuario de Amazon EC2 Auto Scaling.

- Para obtener más información sobre la API, consulte PutScheduledUpdateGroupActionla Referencia de AWS CLI comandos.

PowerShell

Herramientas para la PowerShell versión 4

Ejemplo 1: En este ejemplo se crea o actualiza una acción programada única para cambiar la capacidad deseada a la hora de inicio especificada.

```
Write-ASScheduledUpdateGroupAction -AutoScalingGroupName my-asg -
ScheduledActionName "myScheduledAction" -StartTime "2015-12-01T00:00:00Z" -
DesiredCapacity 10
```

- Para obtener más información sobre la API, consulte PutScheduledUpdateGroupAction Herramientas de AWS para PowerShellCmdlet Reference (V4).

Herramientas para la versión 5 PowerShell

Ejemplo 1: En este ejemplo se crea o actualiza una acción programada única para cambiar la capacidad deseada a la hora de inicio especificada.

```
Write-ASScheduledUpdateGroupAction -AutoScalingGroupName my-asg -
ScheduledActionName "myScheduledAction" -StartTime "2015-12-01T00:00:00Z" -
DesiredCapacity 10
```

- Para obtener más información sobre la API, consulte PutScheduledUpdateGroupActionla referencia de Herramientas de AWS para PowerShell cmdlets (V5).

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. Utilización de este servicio con un AWS SDK En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Utilizar **`RecordLifecycleActionHeartbeat`** con una CLI

Los siguientes ejemplos de código muestran cómo utilizar `RecordLifecycleActionHeartbeat`.

CLI

AWS CLI

Cómo registrar el latido de una acción del ciclo de vida

En este ejemplo, se registra el latido de una acción del ciclo de vida para mantener la instancia en estado pendiente.

```
aws autoscaling record-lifecycle-action-heartbeat \
    --lifecycle-hook-name my-launch-hook \
    --auto-scaling-group-name my-asg \
    --lifecycle-action-token bcd2f1b8-9a78-44d3-8a7a-4dd07d7cf635
```

Este comando no genera ninguna salida.

Para obtener más información, consulte Enlaces de ciclo de vida de Amazon EC2 Auto Scaling en la guía del usuario de Amazon EC2 Auto Scaling.

- Para obtener más información sobre la API, consulte RecordLifecycleActionHeartbeatla Referencia de AWS CLI comandos.

PowerShell

Herramientas para la PowerShell versión 4

Ejemplo 1: En este ejemplo se registra un latido para la acción del ciclo de vida especificada. Esto mantiene la instancia en estado pendiente hasta que complete la acción personalizada.

```
Write-ASLifecycleActionHeartbeat -AutoScalingGroupName my-asg -LifecycleHookName
 myLifecycleHook -LifecycleActionToken bcd2f1b8-9a78-44d3-8a7a-4dd07d7cf635
```

- Para obtener más información sobre la API, consulte RecordLifecycleActionHeartbeat Herramientas de AWS para PowerShellCmdlet Reference (V4).

Herramientas para la versión 5 PowerShell

Ejemplo 1: En este ejemplo se registra un latido para la acción del ciclo de vida especificada. Esto mantiene la instancia en estado pendiente hasta que complete la acción personalizada.

```
Write-ASLifecycleActionHeartbeat -AutoScalingGroupName my-asg -LifecycleHookName
 myLifecycleHook -LifecycleActionToken bcd2f1b8-9a78-44d3-8a7a-4dd07d7cf635
```

- Para obtener más información sobre la API, consulte RecordLifecycleActionHeartbeatla referencia de Herramientas de AWS para PowerShell cmdlets (V5).

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. Utilización de este servicio con un AWS SDK En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Utilizar **`ResumeProcesses`** con una CLI

Los siguientes ejemplos de código muestran cómo utilizar `ResumeProcesses`.

CLI

AWS CLI

Cómo reanudar un proceso suspendido

En este ejemplo, se reanuda el proceso de escalado suspendido especificado para el grupo de escalado automático especificado.

```
aws autoscaling resume-processes \
    --auto-scaling-group-name my-asg \
    --scaling-processes AlarmNotification
```

Este comando no genera ninguna salida.

Para obtener más información, consulte Suspending and resuming scaling processes en la Guía del usuario de Amazon EC2 Auto Scaling.

- Para obtener más información sobre la API, consulte ResumeProcesseslа Referencia de AWS CLI comandos.

PowerShell

Herramientas para la PowerShell versión 4

Ejemplo 1: En este ejemplo se reanuda el proceso de escalado automático especificado para el grupo de escalado automático especificado.

```
Resume-ASProcess -AutoScalingGroupName my-asg -ScalingProcess "AlarmNotification"
```

Ejemplo 2: En este ejemplo se reanudan todos los procesos de escalado automático suspendidos para el grupo de escalado automático especificado.

```
Resume-ASProcess -AutoScalingGroupName my-asg
```

- Para obtener más información sobre la API, consulte ResumeProcesses Herramientas de AWS para PowerShellCmdlet Reference (V4).

Herramientas para la versión 5 PowerShell

Ejemplo 1: En este ejemplo se reanuda el proceso de escalado automático especificado para el grupo de escalado automático especificado.

```
Resume-ASProcess -AutoScalingGroupName my-asg -ScalingProcess "AlarmNotification"
```

Ejemplo 2: En este ejemplo se reanudan todos los procesos de escalado automático suspendidos para el grupo de escalado automático especificado.

```
Resume-ASProcess -AutoScalingGroupName my-asg
```

- Para obtener más información sobre la API, consulte ResumeProcesseslа referencia de Herramientas de AWS para PowerShell cmdlets (V5).

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. Utilización de este servicio con un AWS SDK En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Úselo **`SetDesiredCapacity`** con un AWS SDK o CLI

Los siguientes ejemplos de código muestran cómo utilizar `SetDesiredCapacity`.

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:

- Conceptos básicos

.NET

SDK para .NET (v4)

> **Note**
>
> Hay más información GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
/// <summary>
/// Set the desired capacity of an Auto Scaling group.
/// </summary>
/// <param name="groupName">The name of the Auto Scaling group.</param>
/// <param name="desiredCapacity">The desired capacity for the Auto
/// Scaling group.</param>
```

```
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> SetDesiredCapacityAsync(
        string groupName,
        int desiredCapacity)
    {
        var capacityRequest = new SetDesiredCapacityRequest
        {
            AutoScalingGroupName = groupName,
            DesiredCapacity = desiredCapacity,
        };

        var response = await
_amazonAutoScaling.SetDesiredCapacityAsync(capacityRequest);
        Console.WriteLine($"You have set the DesiredCapacity to
{desiredCapacity}.");

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

- Para obtener más información sobre la API, consulta SetDesiredCapacityla Referencia AWS SDK para .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::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region (overrides config file).
        // clientConfig.region = "us-east-1";

    Aws::AutoScaling::AutoScalingClient autoScalingClient(clientConfig);

        Aws::AutoScaling::Model::SetDesiredCapacityRequest request;
```

```
        request.SetAutoScalingGroupName(groupName);
        request.SetDesiredCapacity(2);

        Aws::AutoScaling::Model::SetDesiredCapacityOutcome outcome =
                autoScalingClient.SetDesiredCapacity(request);

        if (!outcome.IsSuccess()) {
            std::cerr << "Error with AutoScaling::SetDesiredCapacityRequest. "
                      << outcome.GetError().GetMessage()
                      << std::endl;

        }
```

- Para obtener más información sobre la API, consulta SetDesiredCapacityla Referencia AWS SDK para C++ de la API.

CLI

AWS CLI

Fijación de la capacidad deseada inicial para el grupo de escalado automático

En este ejemplo, se establece la capacidad deseada para el grupo de escalado automático especificado.

```
aws autoscaling set-desired-capacity \
    --auto-scaling-group-name my-asg \
    --desired-capacity 2 \
    --honor-cooldown
```

Este comando vuelve a la petición si se ejecuta correctamente.

- Para obtener más información sobre la API, consulta SetDesiredCapacityla 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 setDesiredCapacity(AutoScalingClient autoScalingClient,
String groupName) {
       try {
           SetDesiredCapacityRequest capacityRequest =
SetDesiredCapacityRequest.builder()
                   .autoScalingGroupName(groupName)
                   .desiredCapacity(2)
                   .build();

           autoScalingClient.setDesiredCapacity(capacityRequest);
           System.out.println("You have set the DesiredCapacity to 2");

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

- Para obtener más información sobre la API, consulta SetDesiredCapacityla Referencia AWS SDK for Java 2.x 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 setDesiredCapacity(groupName: String) {
    val capacityRequest =
        SetDesiredCapacityRequest {
            autoScalingGroupName = groupName
            desiredCapacity = 2
        }

    AutoScalingClient { region = "us-east-1" }.use { autoScalingClient ->
        autoScalingClient.setDesiredCapacity(capacityRequest)
        println("You set the DesiredCapacity to 2")
    }
}
```

- Para obtener más información sobre la API, consulta SetDesiredCapacityla referencia sobre el AWS SDK para la API de Kotlin.

PHP

SDK para PHP

> **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 function setDesiredCapacity($autoScalingGroupName, $desiredCapacity)
    {
        return $this->autoScalingClient->setDesiredCapacity([
            'AutoScalingGroupName' => $autoScalingGroupName,
            'DesiredCapacity' => $desiredCapacity,
        ]);
    }
```

- Para obtener más información sobre la API, consulta SetDesiredCapacityla Referencia AWS SDK para PHP de la API.

PowerShell

Herramientas para la PowerShell versión 4

Ejemplo 1: En este ejemplo se establece el tamaño del grupo de escalado automático especificado.

```
Set-ASDesiredCapacity -AutoScalingGroupName my-asg -DesiredCapacity 2
```

Ejemplo 2: En este ejemplo se establece el tamaño del grupo de escalado automático especificado y espera a que finalice el período de recuperación antes de escalar al nuevo tamaño.

```
Set-ASDesiredCapacity -AutoScalingGroupName my-asg -DesiredCapacity 2 -
HonorCooldown $true
```

- Para obtener más información sobre la API, consulte SetDesiredCapacity Herramientas de AWS para PowerShellCmdlet Reference (V4).

Herramientas para la versión 5 PowerShell

Ejemplo 1: En este ejemplo se establece el tamaño del grupo de escalado automático especificado.

```
Set-ASDesiredCapacity -AutoScalingGroupName my-asg -DesiredCapacity 2
```

Ejemplo 2: En este ejemplo se establece el tamaño del grupo de escalado automático especificado y espera a que finalice el período de recuperación antes de escalar al nuevo tamaño.

```
Set-ASDesiredCapacity -AutoScalingGroupName my-asg -DesiredCapacity 2 -
HonorCooldown $true
```

- Para obtener más información sobre la API, consulte SetDesiredCapacityla referencia de Herramientas de AWS para PowerShell cmdlets (V5).

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 AutoScalingWrapper:
    """Encapsulates Amazon EC2 Auto Scaling actions."""

    def __init__(self, autoscaling_client):
        """
        :param autoscaling_client: A Boto3 Amazon EC2 Auto Scaling client.
        """
        self.autoscaling_client = autoscaling_client


    def set_desired_capacity(self, group_name: str, capacity: int) -> None:
        """
        Sets the desired capacity of the group. Amazon EC2 Auto Scaling tries to
 keep the
        number of running instances equal to the desired capacity.

        :param group_name: The name of the group to update.
        :param capacity: The desired number of running instances.
        :return: None
        :raises ClientError: If there is an error setting the desired capacity.
        """
        try:
            self.autoscaling_client.set_desired_capacity(
                AutoScalingGroupName=group_name,
                DesiredCapacity=capacity,
                HonorCooldown=False,
            )
            logger.info(
                f"Successfully set desired capacity of {capacity} for Auto
 Scaling group '{group_name}'."
            )
```

```
        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            logger.error(
                f"Failed to set desired capacity for Auto Scaling group
'{group_name}'."
            )
            if error_code == "ScalingActivityInProgress":
                logger.error(
                    f"A scaling activity is currently in progress for the Auto
Scaling group '{group_name}'. "
                    "Please wait for the activity to complete before attempting
to set the desired capacity."
                )
            logger.error(f"Full error:\n\t{err}")
            raise
```

- Para obtener más información sobre la API, consulta SetDesiredCapacityla 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.

```
   pub async fn scale_desired_capacity(&self, capacity: i32) -> Result<(),
ScenarioError> {
        // 7. SetDesiredCapacity: set desired capacity to 2.
        //   Wait for a second instance to launch.
        let update_group = self
            .autoscaling
            .set_desired_capacity()
            .auto_scaling_group_name(self.auto_scaling_group_name.clone())
            .desired_capacity(capacity)
            .send()
```

```
            .await;
        if let Err(err) = update_group {
            return Err(ScenarioError::new(
                format!("Failed to update group to desired capacity
({capacity})").as_str(),
                &err,
            ));
        }
        Ok(())
    }
```

- Para obtener más información sobre la API, consulta SetDesiredCapacityla referencia sobre la API de AWS SDK para Rust.

SAP ABAP

SDK para 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.

```
" Example: iv_group_name = 'my-auto-scaling-group'
" Example: iv_capacity = 2

TRY.
    ao_asc->setdesiredcapacity(
      iv_autoscalinggroupname = iv_group_name
      iv_desiredcapacity = iv_capacity
      iv_honorcooldown = abap_false ).

    MESSAGE 'Desired capacity set successfully' TYPE 'I'.

  CATCH /aws1/cx_ascscaactivityinprg00 INTO DATA(lo_activity_in_progress).
    RAISE EXCEPTION lo_activity_in_progress.
  CATCH /aws1/cx_rt_generic INTO DATA(lo_generic_exception).
    RAISE EXCEPTION lo_generic_exception.
ENDTRY.
```

- Para obtener más información sobre la API, consulte SetDesiredCapacityla 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, consulteUtilización de este servicio con un AWS SDK. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Utilizar **`SetInstanceHealth`** con una CLI

Los siguientes ejemplos de código muestran cómo utilizar `SetInstanceHealth`.

CLI

AWS CLI

Cómo comprobar el estado de una instancia

En este ejemplo se establece el estado de la instancia especificada en `Unhealthy`.

```
aws autoscaling set-instance-health \
    --instance-id i-061c63c5eb45f0416 \
    --health-status Unhealthy
```

Este comando no genera ninguna salida.

- Para obtener más información sobre la API, consulte SetInstanceHealthla Referencia de AWS CLI comandos.

PowerShell

Herramientas para la PowerShell versión 4

Ejemplo 1: En este ejemplo, se establece el estado de la instancia especificada en «Mal estado», por lo que queda fuera de servicio. El escalado automático finaliza y reemplaza la instancia.

```
Set-ASInstanceHealth -HealthStatus Unhealthy -InstanceId i-93633f9b
```

Ejemplo 2: En este ejemplo, se establece el estado de la instancia especificada en «Buen estado» y se mantiene en servicio. No se respeta ningún periodo de gracia de la comprobación de estado para un grupo de escalado automático.

```
Set-ASInstanceHealth -HealthStatus Healthy -InstanceId i-93633f9b -
ShouldRespectGracePeriod $false
```

- Para obtener más información sobre la API, consulte SetInstanceHealth Herramientas de AWS para PowerShellCmdlet Reference (V4).

Herramientas para la versión 5 PowerShell

Ejemplo 1: En este ejemplo, se establece el estado de la instancia especificada en «Mal estado», por lo que queda fuera de servicio. El escalado automático finaliza y reemplaza la instancia.

```
Set-ASInstanceHealth -HealthStatus Unhealthy -InstanceId i-93633f9b
```

Ejemplo 2: En este ejemplo, se establece el estado de la instancia especificada en «Buen estado» y se mantiene en servicio. No se respeta ningún periodo de gracia de la comprobación de estado para un grupo de escalado automático.

```
Set-ASInstanceHealth -HealthStatus Healthy -InstanceId i-93633f9b -
ShouldRespectGracePeriod $false
```

- Para obtener más información sobre la API, consulte SetInstanceHealthla referencia de Herramientas de AWS para PowerShell cmdlets (V5).

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. Utilización de este servicio con un AWS SDK En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Utilizar **`SetInstanceProtection`** con una CLI

Los siguientes ejemplos de código muestran cómo utilizar `SetInstanceProtection`.

CLI

AWS CLI

Ejemplo 1: Cómo habilitar la configuración de protección de instancias para una instancia

En este ejemplo, se habilita la protección de instancias para la instancia especificada.

```
aws autoscaling set-instance-protection \
    --instance-ids i-061c63c5eb45f0416 \
    --auto-scaling-group-name my-asg --protected-from-scale-in
```

Este comando no genera ninguna salida.

Ejemplo 2: deshabilitar la configuración de protección de instancias para una instancia

En este ejemplo, se deshabilita la protección de instancias para la instancia especificada.

```
aws autoscaling set-instance-protection \
    --instance-ids i-061c63c5eb45f0416 \
    --auto-scaling-group-name my-asg \
    --no-protected-from-scale-in
```

Este comando no genera ninguna salida.

- Para obtener más información sobre la API, consulte SetInstanceProtectionla Referencia de AWS CLI comandos.

PowerShell

Herramientas para la PowerShell versión 4

Ejemplo 1: En este ejemplo se habilita la protección de instancias para la instancia especificada.

```
Set-ASInstanceProtection -AutoScalingGroupName my-asg -InstanceId i-12345678 -
ProtectedFromScaleIn $true
```

Ejemplo 2: En este ejemplo se deshabilita la protección de instancias para la instancia especificada.

```
Set-ASInstanceProtection -AutoScalingGroupName my-asg -InstanceId i-12345678 -
ProtectedFromScaleIn $false
```

- Para obtener más información sobre la API, consulte SetInstanceProtection Herramientas de AWS para PowerShellCmdlet Reference (V4).

Herramientas para la versión 5 PowerShell

Ejemplo 1: En este ejemplo se habilita la protección de instancias para la instancia especificada.

```
Set-ASInstanceProtection -AutoScalingGroupName my-asg -InstanceId i-12345678 -
ProtectedFromScaleIn $true
```

Ejemplo 2: En este ejemplo se deshabilita la protección de instancias para la instancia especificada.

```
Set-ASInstanceProtection -AutoScalingGroupName my-asg -InstanceId i-12345678 -
ProtectedFromScaleIn $false
```

- Para obtener más información sobre la API, consulte SetInstanceProtectionla referencia de Herramientas de AWS para PowerShell cmdlets (V5).

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. Utilización de este servicio con un AWS SDK En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Utilizar **`SuspendProcesses`** con una CLI

Los siguientes ejemplos de código muestran cómo utilizar `SuspendProcesses`.

CLI

AWS CLI

Cómo suspender los procesos de escalado automático

En este ejemplo, se suspende el proceso de escalado especificado para el grupo de escalado automático especificado.

```
aws autoscaling suspend-processes \
    --auto-scaling-group-name my-asg \
    --scaling-processes AlarmNotification
```

Este comando no genera ninguna salida.

Para obtener más información, consulte Suspending and resuming scaling processes en la Guía del usuario de Amazon EC2 Auto Scaling.

- Para obtener más información sobre la API, consulte SuspendProcesses la Referencia de AWS CLI comandos.

PowerShell

Herramientas para la PowerShell versión 4

Ejemplo 1: En este ejemplo se suspende el proceso de escalado automático especificado para el grupo de escalado automático especificado.

```
Suspend-ASProcess -AutoScalingGroupName my-asg -ScalingProcess
 "AlarmNotification"
```

Ejemplo 2: En este ejemplo se suspenden todos los procesos de escalado automático para el grupo de escalado automático especificado.

```
Suspend-ASProcess -AutoScalingGroupName my-asg
```

- Para obtener más información sobre la API, consulte SuspendProcesses Herramientas de AWS para PowerShell Cmdlet Reference (V4).

Herramientas para la versión 5 PowerShell

Ejemplo 1: En este ejemplo se suspende el proceso de escalado automático especificado para el grupo de escalado automático especificado.

```
Suspend-ASProcess -AutoScalingGroupName my-asg -ScalingProcess
 "AlarmNotification"
```

Ejemplo 2: En este ejemplo se suspenden todos los procesos de escalado automático para el grupo de escalado automático especificado.

```
Suspend-ASProcess -AutoScalingGroupName my-asg
```

- Para obtener más información sobre la API, consulte SuspendProcesses la referencia de Herramientas de AWS para PowerShell cmdlets (V5).

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulte. Utilización de este servicio con un AWS SDK En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Úselo **`TerminateInstanceInAutoScalingGroup`** con un AWS SDK o CLI

Los siguientes ejemplos de código muestran cómo utilizar `TerminateInstanceInAutoScalingGroup`.

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:

- Conceptos básicos
- Creación y administración de un servicio resiliente

.NET

SDK para .NET (v4)

> **Note**
>
> Hay más información GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
    /// <summary>
    /// Terminate all instances in the Auto Scaling group in preparation for
    /// deleting the group.
    /// </summary>
    /// <param name="instanceId">The instance Id of the instance to terminate.</
param>
    /// <returns>A Boolean value that indicates the success or failure of
    /// the operation.</returns>
    public async Task<bool> TerminateInstanceInAutoScalingGroupAsync(
        string instanceId)
    {
        var request = new TerminateInstanceInAutoScalingGroupRequest
        {
            InstanceId = instanceId,
            ShouldDecrementDesiredCapacity = false,
```

```
        };

        var response = await
_amazonAutoScaling.TerminateInstanceInAutoScalingGroupAsync(request);

        if (response.HttpStatusCode == System.Net.HttpStatusCode.OK)
        {
            Console.WriteLine($"You have terminated the instance: {instanceId}");
            return true;
        }

        Console.WriteLine($"Could not terminate {instanceId}");
        return false;
    }
```

- Para obtener más información sobre la API, consulta TerminateInstanceInAutoScalingGroupla Referencia AWS SDK para .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::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region (overrides config file).
        // clientConfig.region = "us-east-1";

    Aws::AutoScaling::AutoScalingClient autoScalingClient(clientConfig);

        Aws::AutoScaling::Model::TerminateInstanceInAutoScalingGroupRequest
request;
        request.SetInstanceId(instanceIDs[instanceNumber - 1]);
        request.SetShouldDecrementDesiredCapacity(false);
```

```
        Aws::AutoScaling::Model::TerminateInstanceInAutoScalingGroupOutcome
outcome =
                autoScalingClient.TerminateInstanceInAutoScalingGroup(request);

        if (outcome.IsSuccess()) {
            std::cout << "Waiting for EC2 instance with ID '"
                      << instanceIDs[instanceNumber - 1] << "' to terminate..."
                      << std::endl;
        }
        else {
            std::cerr << "Error with
AutoScaling::TerminateInstanceInAutoScalingGroup. "
                      << outcome.GetError().GetMessage()
                      << std::endl;

        }
```

- Para obtener más información sobre la API, consulta TerminateInstanceInAutoScalingGroupla Referencia AWS SDK para C++ de la API.

CLI

AWS CLI

Finalización de una instancia en su grupo de escalado automático

En este ejemplo, se termina la instancia especificada del grupo de escalado automático especificado sin actualizar el tamaño del grupo. Amazon EC2 Auto Scaling lanza una instancia de remplazo una vez que finalice la instancia especificada.

```
aws autoscaling terminate-instance-in-auto-scaling-group \
    --instance-id i-061c63c5eb45f0416 \
    --no-should-decrement-desired-capacity
```

Salida:

```
{
    "Activities": [
        {
            "ActivityId": "8c35d601-793c-400c-fcd0-f64a27530df7",
            "AutoScalingGroupName": "my-asg",
```

```
            "Description": "Terminating EC2 instance: i-061c63c5eb45f0416",
            "Cause": "",
            "StartTime": "2020-10-31T20:34:25.680Z",
            "StatusCode": "InProgress",
            "Progress": 0,
            "Details": "{\"Subnet ID\":\"subnet-6194ea3b\",\"Availability Zone\":
\"us-west-2c\"}"
        }
    ]
}
```

- Para obtener más información sobre la API, consulta TerminateInstanceInAutoScalingGroupla 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 terminateInstanceInAutoScalingGroup(AutoScalingClient
autoScalingClient, String instanceId) {
       try {
           TerminateInstanceInAutoScalingGroupRequest request =
TerminateInstanceInAutoScalingGroupRequest.builder()
                   .instanceId(instanceId)
                   .shouldDecrementDesiredCapacity(false)
                   .build();

           autoScalingClient.terminateInstanceInAutoScalingGroup(request);
           System.out.println("You have terminated instance " + instanceId);

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

- Para obtener más información sobre la API, consulta TerminateInstanceInAutoScalingGroupla Referencia AWS SDK for Java 2.x 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 terminateInstanceInAutoScalingGroup(instanceIdVal: String) {
    val request =
        TerminateInstanceInAutoScalingGroupRequest {
            instanceId = instanceIdVal
            shouldDecrementDesiredCapacity = false
        }

    AutoScalingClient { region = "us-east-1" }.use { autoScalingClient ->
        autoScalingClient.terminateInstanceInAutoScalingGroup(request)
        println("You have terminated instance $instanceIdVal")
    }
}
```

- Para obtener más información sobre la API, consulta TerminateInstanceInAutoScalingGroupla referencia sobre el AWS SDK para la API de Kotlin.

PHP

SDK para PHP

> **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 function terminateInstanceInAutoScalingGroup(
        $instanceId,
        $shouldDecrementDesiredCapacity = true,
        $attempts = 0
    ) {
        try {
            return $this->autoScalingClient-
>terminateInstanceInAutoScalingGroup([
                'InstanceId' => $instanceId,
                'ShouldDecrementDesiredCapacity' =>
 $shouldDecrementDesiredCapacity,
            ]);
        } catch (AutoScalingException $exception) {
            if ($exception->getAwsErrorCode() == "ScalingActivityInProgress" &&
 $attempts < 5) {
                error_log("Cannot terminate an instance while it is still
 pending. Waiting then trying again.");
                sleep(5 * (1 + $attempts));
                return $this->terminateInstanceInAutoScalingGroup(
                    $instanceId,
                    $shouldDecrementDesiredCapacity,
                    ++$attempts
                );
            } else {
                throw $exception;
            }
        }
    }
```

- Para obtener más información sobre la API, consulta TerminateInstanceInAutoScalingGroupla Referencia AWS SDK para PHP de la API.

PowerShell

Herramientas para la PowerShell versión 4

Ejemplo 1: En este ejemplo, se termina la instancia especificada y se reduce la capacidad deseada de su grupo de escalado automático para que Auto Scaling no inicialice una instancia de sustitución.

```
Stop-ASInstanceInAutoScalingGroup -InstanceId i-93633f9b -
ShouldDecrementDesiredCapacity $true
```

Salida:

```
ActivityId           : 2e40d9bd-1902-444c-abf3-6ea0002efdc5
AutoScalingGroupName :
Cause                : At 2015-11-22T16:09:03Z instance i-93633f9b was taken out
 of service in response to a user
                       request, shrinking the capacity from 2 to 1.
Description          : Terminating EC2 instance: i-93633f9b
Details              : {"Availability Zone":"us-west-2b","Subnet
 ID":"subnet-5264e837"}
EndTime              :
Progress             : 0
StartTime            : 11/22/2015 8:09:03 AM
StatusCode           : InProgress
StatusMessage        :
```

Ejemplo 2: En este ejemplo, se termina la instancia especificada sin disminuir la capacidad deseada de su grupo de escalado automático. Auto Scaling inicializa una instancia de remplazo.

```
Stop-ASInstanceInAutoScalingGroup -InstanceId i-93633f9b -
ShouldDecrementDesiredCapacity $false
```

Salida:

```
ActivityId           : 2e40d9bd-1902-444c-abf3-6ea0002efdc5
AutoScalingGroupName :
Cause                : At 2015-11-22T16:09:03Z instance i-93633f9b was taken out
 of service in response to a user
                       request.
Description          : Terminating EC2 instance: i-93633f9b
```

```
Details              : {"Availability Zone":"us-west-2b","Subnet
 ID":"subnet-5264e837"}
EndTime              :
Progress             : 0
StartTime            : 11/22/2015 8:09:03 AM
StatusCode           : InProgress
StatusMessage        :
```

- Para obtener más información sobre la API, consulte TerminateInstanceInAutoScalingGroup Herramientas de AWS para PowerShellCmdlet Reference (V4).

Herramientas para la versión 5 PowerShell

Ejemplo 1: En este ejemplo, se termina la instancia especificada y se reduce la capacidad deseada de su grupo de escalado automático para que Auto Scaling no inicialice una instancia de sustitución.

```
Stop-ASInstanceInAutoScalingGroup -InstanceId i-93633f9b -
ShouldDecrementDesiredCapacity $true
```

Salida:

```
ActivityId           : 2e40d9bd-1902-444c-abf3-6ea0002efdc5
AutoScalingGroupName :
Cause                : At 2015-11-22T16:09:03Z instance i-93633f9b was taken out
 of service in response to a user
                       request, shrinking the capacity from 2 to 1.
Description          : Terminating EC2 instance: i-93633f9b
Details              : {"Availability Zone":"us-west-2b","Subnet
 ID":"subnet-5264e837"}
EndTime              :
Progress             : 0
StartTime            : 11/22/2015 8:09:03 AM
StatusCode           : InProgress
StatusMessage        :
```

Ejemplo 2: En este ejemplo, se termina la instancia especificada sin disminuir la capacidad deseada de su grupo de escalado automático. Auto Scaling inicializa una instancia de remplazo.

```
Stop-ASInstanceInAutoScalingGroup -InstanceId i-93633f9b -
ShouldDecrementDesiredCapacity $false
```

Salida:

```
ActivityId           : 2e40d9bd-1902-444c-abf3-6ea0002efdc5
AutoScalingGroupName :
Cause                : At 2015-11-22T16:09:03Z instance i-93633f9b was taken out
 of service in response to a user
                       request.
Description          : Terminating EC2 instance: i-93633f9b
Details              : {"Availability Zone":"us-west-2b","Subnet
 ID":"subnet-5264e837"}
EndTime              :
Progress             : 0
StartTime            : 11/22/2015 8:09:03 AM
StatusCode           : InProgress
StatusMessage        :
```

- Para obtener más información sobre la API, consulte TerminateInstanceInAutoScalingGroupla referencia de Herramientas de AWS para PowerShell cmdlets (V5).

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 AutoScalingWrapper:
    """Encapsulates Amazon EC2 Auto Scaling actions."""

    def __init__(self, autoscaling_client):
        """
        :param autoscaling_client: A Boto3 Amazon EC2 Auto Scaling client.
        """
        self.autoscaling_client = autoscaling_client


    def terminate_instance(
```

```
        self, instance_id: str, decrease_capacity: bool
    ) -> Dict[str, Any]:
        """
        Stops an instance.

        :param instance_id: The ID of the instance to stop.
        :param decrease_capacity: Specifies whether to decrease the desired
capacity
                                  of the group. When passing True for this
parameter,
                                  you can stop an instance without having a
replacement
                                  instance start when the desired capacity
threshold is
                                  crossed.
        :return: A dictionary containing details of the scaling activity that
occurs
                 in response to this action.
        :raises ClientError: If there is an error terminating the instance.
        """
        try:
            response =
self.autoscaling_client.terminate_instance_in_auto_scaling_group(
                InstanceId=instance_id,
ShouldDecrementDesiredCapacity=decrease_capacity
            )
            logger.info(f"Successfully terminated instance {instance_id}.")
            return response["Activity"]

        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            logger.error(f"Failed to terminate instance {instance_id}.")
            if error_code == "ScalingActivityInProgress":
                logger.error(
                    "A scaling activity is currently in progress for the Auto
Scaling group "
                    f"associated with instance '{instance_id}'. "
                    "Please wait for the activity to complete before attempting
to terminate the instance."
                )
            elif error_code == "ResourceInUse":
                logger.error(
                    f"The instance '{instance_id}' or an associated resource is
currently in use "
```

```
                    "and cannot be terminated. "
                    "Ensure the instance is not involved in any ongoing processes
and try again."
                )
            logger.error(f"Full error:\n\t{err}")
            raise
```

- Para obtener más información sobre la API, consulta [TerminateInstanceInAutoScalingGroup](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.

```
    pub async fn terminate_some_instance(&self) -> Result<(), ScenarioError> {
        // Retrieve a list of instances in the auto scaling group.
        let auto_scaling_group = self.get_group().await?;
        let instances = auto_scaling_group.instances();
        // Or use other logic to find an instance to terminate.
        let instance = instances.first();
        if let Some(instance) = instance {
            let instance_id = if let Some(instance_id) = instance.instance_id() {
                instance_id
            } else {
                return Err(ScenarioError::with("Missing instance id"));
            };
            let termination = self
                .ec2
                .terminate_instances()
                .instance_ids(instance_id)
                .send()
                .await;
```

```
            if let Err(err) = termination {
                Err(ScenarioError::new(
                    "There was a problem terminating an instance",
                    &err,
                ))
            } else {
                Ok(())
            }
        } else {
            Err(ScenarioError::with("There was no instance to terminate"))
        }
    }

    async fn get_group(&self) -> Result<AutoScalingGroup, ScenarioError> {
        let describe_auto_scaling_groups = self
            .autoscaling
            .describe_auto_scaling_groups()
            .auto_scaling_group_names(self.auto_scaling_group_name.clone())
            .send()
            .await;

        if let Err(err) = describe_auto_scaling_groups {
            return Err(ScenarioError::new(
                format!(
                    "Failed to get status of autoscaling group {}",
                    self.auto_scaling_group_name.clone()
                )
                .as_str(),
                &err,
            ));
        }

        let describe_auto_scaling_groups_output =
describe_auto_scaling_groups.unwrap();
        let auto_scaling_groups =
describe_auto_scaling_groups_output.auto_scaling_groups();
        let auto_scaling_group = auto_scaling_groups.first();

        if auto_scaling_group.is_none() {
            return Err(ScenarioError::with(format!(
                "Could not find autoscaling group {}",
                self.auto_scaling_group_name.clone()
            )));
        }
```

```
    Ok(auto_scaling_group.unwrap().clone())
}
```

- Para obtener más información sobre la API, consulta TerminateInstanceInAutoScalingGroupla referencia sobre la API de AWS SDK para Rust.

SAP ABAP

SDK para 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.

```
" Example: iv_instance_id = 'i-1234567890abcdef0'
" Example: iv_decrease_capacity = abap_true

TRY.
    DATA(lo_output) = ao_asc->terminateinstinautoscgroup(
      iv_instanceid = iv_instance_id
      iv_shoulddecrementdesiredcap = iv_decrease_capacity ).

    oo_output = lo_output->get_activity( ).

    MESSAGE 'Instance terminated successfully' TYPE 'I'.

  CATCH /aws1/cx_ascscaactivityinprg00 INTO DATA(lo_activity_in_progress).
    RAISE EXCEPTION lo_activity_in_progress.
  CATCH /aws1/cx_ascresrccontionfault INTO DATA(lo_contention).
    RAISE EXCEPTION lo_contention.
  CATCH /aws1/cx_rt_generic INTO DATA(lo_generic_exception).
    RAISE EXCEPTION lo_generic_exception.
ENDTRY.
```

- Para obtener más información sobre la API, consulte TerminateInstanceInAutoScalingGroupla 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, consulteUtilización de este servicio con un AWS SDK. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Úselo **`UpdateAutoScalingGroup`** con un AWS SDK o CLI

Los siguientes ejemplos de código muestran cómo utilizar `UpdateAutoScalingGroup`.

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:

- Conceptos básicos
- Creación y administración de un servicio resiliente

.NET

SDK para .NET (v4)

> **Note**
>
> Hay más información GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
    /// <summary>
    /// Update the capacity of an Auto Scaling group.
    /// </summary>
    /// <param name="groupName">The name of the Auto Scaling group.</param>
    /// <param name="launchTemplateName">The name of the EC2 launch template.</
param>
    /// <param name="maxSize">The maximum number of instances that can be
    /// created for the Auto Scaling group.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> UpdateAutoScalingGroupAsync(
        string groupName,
        string launchTemplateName,
```

```
        int maxSize)
    {
        var templateSpecification = new LaunchTemplateSpecification
        {
            LaunchTemplateName = launchTemplateName,
        };

        var groupRequest = new UpdateAutoScalingGroupRequest
        {
            MaxSize = maxSize,
            AutoScalingGroupName = groupName,
            LaunchTemplate = templateSpecification,
        };

        var response = await
_amazonAutoScaling.UpdateAutoScalingGroupAsync(groupRequest);
        if (response.HttpStatusCode == System.Net.HttpStatusCode.OK)
        {
            Console.WriteLine($"You successfully updated the Auto Scaling group
{groupName}.");
            return true;
        }
        else
        {
            return false;
        }
    }
```

- Para obtener más información sobre la API, consulta UpdateAutoScalingGroupla Referencia AWS SDK para .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::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region (overrides config file).
        // clientConfig.region = "us-east-1";

    Aws::AutoScaling::AutoScalingClient autoScalingClient(clientConfig);

        Aws::AutoScaling::Model::UpdateAutoScalingGroupRequest request;
        request.SetAutoScalingGroupName(groupName);
        request.SetMaxSize(3);

        Aws::AutoScaling::Model::UpdateAutoScalingGroupOutcome outcome =
                autoScalingClient.UpdateAutoScalingGroup(request);

        if (!outcome.IsSuccess()) {
            std::cerr << "Error with AutoScaling::UpdateAutoScalingGroup. "
                      << outcome.GetError().GetMessage()
                      << std::endl;

        }
```

- Para obtener más información sobre la API, consulta UpdateAutoScalingGroupla Referencia AWS SDK para C++ de la API.

CLI

AWS CLI

Ejemplo 1: Actualización de los límites de tamaño de un grupo de escalado automático

En este ejemplo, se actualiza el grupo de escalado automático especificado con un tamaño mínimo de 2 y un tamaño máximo de 10.

```
aws autoscaling update-auto-scaling-group \
    --auto-scaling-group-name my-asg \
    --min-size 2 \
    --max-size 10
```

Este comando no genera ninguna salida.

Para obtener más información, consulte Establecer límites de capacidad para su grupo de escalado automático en la Guía del usuario de Amazon EC2 Auto Scaling.

Ejemplo 2: Adición de comprobaciones de estado de Elastic Load Balancing y especificar qué zonas de disponibilidad y subredes se deben utilizar

En este ejemplo, se actualiza el grupo de escalado automático especificado para añadir comprobaciones de estado de Elastic Load Balancing. Este comando también actualiza el valor de `--vpc-zone-identifier` con una lista de subredes IDs en varias zonas de disponibilidad.

```
aws autoscaling update-auto-scaling-group \
    --auto-scaling-group-name my-asg \
    --health-check-type ELB \
    --health-check-grace-period 600 \
    --vpc-zone-identifier "subnet-5ea0c127,subnet-6194ea3b,subnet-c934b782"
```

Este comando no genera ninguna salida.

Para obtener más información, consulte Elastic Load Balancing y Amazon EC2 Auto Scaling en la Guía del usuario de Amazon EC2 Auto Scaling.

Ejemplo 3: actualización del grupo con ubicación y la política de terminación

En este ejemplo, se actualizan el grupo con ubicación y la política de terminación que se van a utilizar.

```
aws autoscaling update-auto-scaling-group \
    --auto-scaling-group-name my-asg \
    --placement-group my-placement-group \
    --termination-policies "OldestInstance"
```

Este comando no genera ninguna salida.

Para obtener más información, consulte Grupos de escalado automático en la Guía del usuario de Amazon EC2 Auto Scaling.

Ejemplo 4: uso de la versión más reciente de la plantilla de lanzamiento

En este ejemplo, se actualiza el grupo de escalado automático para utilizar la versión más reciente de la plantilla de lanzamiento.

```
aws autoscaling update-auto-scaling-group \
    --auto-scaling-group-name my-asg \
    --launch-template LaunchTemplateId=lt-1234567890abcde12,Version='$Latest'
```

Este comando no genera ninguna salida.

Para obtener más información, consulte Plantillas de lanzamiento en la Guía del usuario de Amazon EC2 Auto Scaling.

Ejemplo 5: uso de una versión específica de la plantilla de lanzamiento

En este ejemplo, se actualiza el grupo de escalado automático especificado para utilizar una versión específica de una plantilla de lanzamiento en lugar de la versión más reciente o predeterminada.

```
aws autoscaling update-auto-scaling-group \
    --auto-scaling-group-name my-asg \
    --launch-template LaunchTemplateName=my-template-for-auto-scaling,Version='2'
```

Este comando no genera ninguna salida.

Para obtener más información, consulte Plantillas de lanzamiento en la Guía del usuario de Amazon EC2 Auto Scaling.

Ejemplo 6: definición de una política de instancias mixtas y habilitar el reequilibrio de la capacidad

En este ejemplo, se actualiza el grupo de escalado automático especificado para que utilice una política de instancias mixtas y se habilita el reequilibrio de la capacidad. Esta estructura le permite especificar grupos con capacidades bajo demanda y de spot y utilizar distintas plantillas de lanzamiento para diferentes arquitecturas.

```
aws autoscaling update-auto-scaling-group \
    --cli-input-json file://~/config.json
```

Contenido de `config.json`:

```
{
    "AutoScalingGroupName": "my-asg",
    "CapacityRebalance": true,
    "MixedInstancesPolicy": {
        "LaunchTemplate": {
            "LaunchTemplateSpecification": {
                "LaunchTemplateName": "my-launch-template-for-x86",
                "Version": "$Latest"
            },
```

```
            "Overrides": [
                {
                    "InstanceType": "c6g.large",
                    "LaunchTemplateSpecification": {
                        "LaunchTemplateName": "my-launch-template-for-arm",
                        "Version": "$Latest"
                    }
                },
                {
                    "InstanceType": "c5.large"
                },
                {
                    "InstanceType": "c5a.large"
                }
            ]
        },
        "InstancesDistribution": {
            "OnDemandPercentageAboveBaseCapacity": 50,
            "SpotAllocationStrategy": "capacity-optimized"
        }
    }
}
```

Este comando no genera ninguna salida.

Para obtener más información, consulte la sección sobre Grupos de escalado automático con varios tipos de instancia y opciones de compra en la guía del usuario de Amazon EC2 Auto Scaling.

- Para obtener más información sobre la API, consulte UpdateAutoScalingGroupla 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 updateAutoScalingGroup(AutoScalingClient
autoScalingClient, String groupName,
           String launchTemplateName) {
       try {
           AutoScalingWaiter waiter = autoScalingClient.waiter();
           LaunchTemplateSpecification templateSpecification =
LaunchTemplateSpecification.builder()
                   .launchTemplateName(launchTemplateName)
                   .build();

           UpdateAutoScalingGroupRequest groupRequest =
UpdateAutoScalingGroupRequest.builder()
                   .maxSize(3)
                   .autoScalingGroupName(groupName)
                   .launchTemplate(templateSpecification)
                   .build();

           autoScalingClient.updateAutoScalingGroup(groupRequest);
           DescribeAutoScalingGroupsRequest groupsRequest =
DescribeAutoScalingGroupsRequest.builder()
                   .autoScalingGroupNames(groupName)
                   .build();

           WaiterResponse<DescribeAutoScalingGroupsResponse> waiterResponse =
waiter
                   .waitUntilGroupInService(groupsRequest);
           waiterResponse.matched().response().ifPresent(System.out::println);
           System.out.println("You successfully updated the auto scaling group
" + groupName);

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

- Para obtener más información sobre la API, consulta UpdateAutoScalingGroupla Referencia AWS SDK for Java 2.x 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 updateAutoScalingGroup(
    groupName: String,
    launchTemplateNameVal: String,
    serviceLinkedRoleARNVal: String,
) {
    val templateSpecification =
        LaunchTemplateSpecification {
            launchTemplateName = launchTemplateNameVal
        }

    val groupRequest =
        UpdateAutoScalingGroupRequest {
            maxSize = 3
            serviceLinkedRoleArn = serviceLinkedRoleARNVal
            autoScalingGroupName = groupName
            launchTemplate = templateSpecification
        }

    val groupsRequestWaiter =
        DescribeAutoScalingGroupsRequest {
            autoScalingGroupNames = listOf(groupName)
        }

    AutoScalingClient { region = "us-east-1" }.use { autoScalingClient ->
        autoScalingClient.updateAutoScalingGroup(groupRequest)
        autoScalingClient.waitUntilGroupExists(groupsRequestWaiter)
        println("You successfully updated the Auto Scaling group  $groupName")
    }
}
```

- Para obtener más información sobre la API, consulta UpdateAutoScalingGroupla referencia sobre el AWS SDK para la API de Kotlin.

PHP

SDK para PHP

> **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 function updateAutoScalingGroup($autoScalingGroupName, $args)
{
    if (array_key_exists('MaxSize', $args)) {
        $maxSize = ['MaxSize' => $args['MaxSize']];
    } else {
        $maxSize = [];
    }
    if (array_key_exists('MinSize', $args)) {
        $minSize = ['MinSize' => $args['MinSize']];
    } else {
        $minSize = [];
    }
    $parameters = ['AutoScalingGroupName' => $autoScalingGroupName];
    $parameters = array_merge($parameters, $minSize, $maxSize);
    return $this->autoScalingClient->updateAutoScalingGroup($parameters);
}
```

- Para obtener más información sobre la API, consulta UpdateAutoScalingGroupla Referencia AWS SDK para PHP de la API.

PowerShell

Herramientas para la PowerShell versión 4

Ejemplo 1: En este ejemplo, se actualiza el tamaño mínimo y máximo del grupo de escalado automático especificado.

```
Update-ASAutoScalingGroup -AutoScalingGroupName my-asg -MaxSize 5 -MinSize 1
```

Ejemplo 2: En este ejemplo se actualiza el período de recuperación predeterminado del grupo de escalado automático especificado.

```
Update-ASAutoScalingGroup -AutoScalingGroupName my-asg -DefaultCooldown 10
```

Ejemplo 3: En este ejemplo se actualizan las zonas de disponibilidad del grupo de escalado automático especificado.

```
Update-ASAutoScalingGroup -AutoScalingGroupName my-asg -AvailabilityZone @("us-
west-2a", "us-west-2b")
```

Ejemplo 4: En este ejemplo, se actualiza el grupo de escalado automático especificado para utilizar comprobaciones de estado de Elastic Load Balancing.

```
Update-ASAutoScalingGroup -AutoScalingGroupName my-asg -HealthCheckType ELB -
HealthCheckGracePeriod 60
```

- Para obtener más información sobre la API, consulte UpdateAutoScalingGroup Herramientas de AWS para PowerShellCmdlet Reference (V4).

Herramientas para la versión 5 PowerShell

Ejemplo 1: En este ejemplo, se actualiza el tamaño mínimo y máximo del grupo de escalado automático especificado.

```
Update-ASAutoScalingGroup -AutoScalingGroupName my-asg -MaxSize 5 -MinSize 1
```

Ejemplo 2: En este ejemplo se actualiza el período de recuperación predeterminado del grupo de escalado automático especificado.

```
Update-ASAutoScalingGroup -AutoScalingGroupName my-asg -DefaultCooldown 10
```

Ejemplo 3: En este ejemplo se actualizan las zonas de disponibilidad del grupo de escalado automático especificado.

```
Update-ASAutoScalingGroup -AutoScalingGroupName my-asg -AvailabilityZone @("us-
west-2a", "us-west-2b")
```

Ejemplo 4: En este ejemplo, se actualiza el grupo de escalado automático especificado para utilizar comprobaciones de estado de Elastic Load Balancing.

```
Update-ASAutoScalingGroup -AutoScalingGroupName my-asg -HealthCheckType ELB -
HealthCheckGracePeriod 60
```

- Para obtener más información sobre la API, consulte UpdateAutoScalingGroupla referencia de Herramientas de AWS para PowerShell cmdlets (V5).

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 AutoScalingWrapper:
    """Encapsulates Amazon EC2 Auto Scaling actions."""

    def __init__(self, autoscaling_client):
        """
        :param autoscaling_client: A Boto3 Amazon EC2 Auto Scaling client.
        """
        self.autoscaling_client = autoscaling_client


    def update_group(self, group_name: str, **kwargs: Any) -> None:
        """
        Updates an Auto Scaling group.

        :param group_name: The name of the group to update.
        :param kwargs: Keyword arguments to pass through to the service.
        :return: None
        :raises ClientError: If there is an error updating the Auto Scaling
 group.
        """
        try:
            self.autoscaling_client.update_auto_scaling_group(
```

```
                AutoScalingGroupName=group_name, **kwargs
            )
            logger.info(f"Successfully updated Auto Scaling group {group_name}.")

        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            logger.error(f"Failed to update Auto Scaling group {group_name}.")
            if error_code == "ResourceInUse":
                logger.error(
                    "The Auto Scaling group '%s' is currently in use and cannot
 be modified. Please try again later.",
                    group_name,
                )
            elif error_code == "ScalingActivityInProgress":
                logger.error(
                    f"A scaling activity is currently in progress for the Auto
 Scaling group '{group_name}'."
                    "Please wait for the activity to complete before attempting
 to update the group."
                )
            logger.error(f"Full error:\n\t{err}")
            raise
```

- Para obtener más información sobre la API, consulta UpdateAutoScalingGroupla 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 update_group(client: &Client, name: &str, size: i32) -> Result<(),
 Error> {
    client
```

```
        .update_auto_scaling_group()
        .auto_scaling_group_name(name)
        .max_size(size)
        .send()
        .await?;

    println!("Updated AutoScaling group");

    Ok(())
}
```

- Para obtener más información sobre la API, consulta UpdateAutoScalingGroupla referencia sobre la API de AWS SDK para Rust.

SAP ABAP

SDK para 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.

```
    " Example: iv_group_name = 'my-auto-scaling-group'
    " Example: iv_max_size = 5

    TRY.
        ao_asc->updateautoscalinggroup(
          iv_autoscalinggroupname = iv_group_name
          iv_maxsize = iv_max_size
          iv_minsize = iv_min_size ).

        MESSAGE 'Auto Scaling group updated successfully' TYPE 'I'.

      CATCH /aws1/cx_ascresrccontionfault INTO DATA(lo_contention).
        RAISE EXCEPTION lo_contention.
      CATCH /aws1/cx_ascscaactivityinprg00 INTO DATA(lo_activity_in_progress).
        RAISE EXCEPTION lo_activity_in_progress.
      CATCH /aws1/cx_rt_generic INTO DATA(lo_generic_exception).
```

```
        RAISE EXCEPTION lo_generic_exception.
    ENDTRY.
```

- Para obtener más información sobre la API, consulte UpdateAutoScalingGroupla 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, consulteUtilización de este servicio con un AWS SDK. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Escenarios para Auto Scaling usando AWS SDKs

Los siguientes ejemplos de código muestran cómo implementar escenarios comunes en Auto Scaling with AWS SDKs. Estas situaciones muestran cómo llevar a cabo tareas específicas con llamadas a varias funciones dentro de Auto Scaling o en combinación con otros Servicios de AWS. En cada escenario se incluye un enlace al código fuente completo, con instrucciones de configuración y ejecución del código.

Los escenarios requieren un nivel intermedio de experiencia para entender las acciones de servicio en su contexto.

Ejemplos

- Cree y gestione un servicio flexible mediante un AWS SDK

Cree y gestione un servicio flexible 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

SDK para .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.TargetGro
 protocol, port, defaultVpc.VpcId);

        await
 _elasticLoadBalancerWrapper.CreateLoadBalancerAndListener(_elasticLoadBalancerWrapper.Lo
 subnetIds, targetGroup);
        await
 _autoScalerWrapper.AttachLoadBalancerToGroup(_autoScalerWrapper.GroupName,
 targetGroup.TargetGroupArn);
        Console.WriteLine("\nVerifying access to the load balancer endpoint...");
        var endPoint = await
 _elasticLoadBalancerWrapper.GetEndpointForLoadBalancerByName(_elasticLoadBalancerWrapper
        var loadBalancerAccess = await
 _elasticLoadBalancerWrapper.VerifyLoadBalancerEndpoint(endPoint);

        if (!loadBalancerAccess)
        {
            Console.WriteLine("\nCouldn't connect to the load balancer, verifying
 that the port is open...");

            var ipString = await _httpClient.GetStringAsync("https://
checkip.amazonaws.com");
            ipString = ipString.Trim();

            var defaultSecurityGroup = await
 _autoScalerWrapper.GetDefaultSecurityGroupForVpc(defaultVpc);
            var portIsOpen =
 _autoScalerWrapper.VerifyInboundPortForGroup(defaultSecurityGroup, port,
 ipString);
            var sshPortIsOpen =
 _autoScalerWrapper.VerifyInboundPortForGroup(defaultSecurityGroup, sshPort,
 ipString);
```

```
            if (!portIsOpen)
            {
                Console.WriteLine(
                    "\nFor this example to work, the default security group for
your default VPC must\n"
                    + "allows access from this computer. You can either add it
automatically from this\n"
                    + "example or add it yourself using the AWS Management
Console.\n");

                if (!interactive || GetYesNoResponse(
                        "Do you want to add a rule to the security group to allow
inbound traffic from your computer's IP address?"))
                {
                    await
_autoScalerWrapper.OpenInboundPort(defaultSecurityGroup.GroupId, port,
ipString);
                }
            }

            if (!sshPortIsOpen)
            {
                if (!interactive || GetYesNoResponse(
                        "Do you want to add a rule to the security group to allow
inbound SSH traffic for debugging from your computer's IP address?"))
                {
                    await
_autoScalerWrapper.OpenInboundPort(defaultSecurityGroup.GroupId, sshPort,
ipString);
                }
            }
            loadBalancerAccess = await
_elasticLoadBalancerWrapper.VerifyLoadBalancerEndpoint(endPoint);
        }

        if (loadBalancerAccess)
        {
            Console.WriteLine("Your load balancer is ready. You can access it by
browsing to:");
            Console.WriteLine($"\thttp://{endPoint}\n");
        }
        else
        {
```

```
            Console.WriteLine(
                "\nCouldn't get a successful response from the load balancer
endpoint. Troubleshoot by\n"
                + "manually verifying that your VPC and security group are
configured correctly and that\n"
                + "you can successfully make a GET request to the load balancer
endpoint:\n");
            Console.WriteLine($"\thttp://{endPoint}\n");
        }
        Console.WriteLine(new string('-', 80));
        Console.WriteLine("Press Enter when you're ready to continue with the
demo.");
        if (interactive)
            Console.ReadLine();
        return true;
    }

    /// <summary>
    /// Demonstrate the steps of the scenario.
    /// </summary>
    /// <param name="interactive">True to run as an interactive scenario.</param>
    /// <returns>Async task.</returns>
    public static async Task<bool> Demo(bool interactive)
    {
        var ssmOnlyPolicy = Path.Join(_configuration["resourcePath"],
            "ssm_only_policy.json");

        Console.WriteLine(new string('-', 80));
        Console.WriteLine("Resetting parameters to starting values for demo.");
        await _smParameterWrapper.Reset();

        Console.WriteLine("\nThis part of the demonstration shows how to toggle
different parts of the system\n" +
                          "to create situations where the web service fails, and
shows how using a resilient\n" +
                          "architecture can keep the web service running in spite
of these failures.");
        Console.WriteLine(new string('-', 88));
        Console.WriteLine("At the start, the load balancer endpoint returns
recommendations and reports that all targets are healthy.");
        if (interactive)
            await DemoActionChoices();
```

```
        Console.WriteLine($"The web service running on the EC2 instances gets
recommendations by querying a DynamoDB table.\n" +
                          $"The table name is contained in a Systems Manager
parameter named '{_smParameterWrapper.TableParameter}'.\n" +
                          $"To simulate a failure of the recommendation service,
let's set this parameter to name a non-existent table.\n");
        await
_smParameterWrapper.PutParameterByName(_smParameterWrapper.TableParameter,
"this-is-not-a-table");
        Console.WriteLine("\nNow, sending a GET request to the load balancer
endpoint returns a failure code. But, the service reports as\n" +
                          "healthy to the load balancer because shallow health
checks don't check for failure of the recommendation service.");
        if (interactive)
            await DemoActionChoices();

        Console.WriteLine("Instead of failing when the recommendation service
fails, the web service can return a static response.");
        Console.WriteLine("While this is not a perfect solution, it presents the
customer with a somewhat better experience than failure.");

        await
_smParameterWrapper.PutParameterByName(_smParameterWrapper.FailureResponseParameter,
"static");

        Console.WriteLine("\nNow, sending a GET request to the load balancer
endpoint returns a static response.");
        Console.WriteLine("The service still reports as healthy because health
checks are still shallow.");
        if (interactive)
            await DemoActionChoices();

        Console.WriteLine("Let's reinstate the recommendation service.\n");
        await
_smParameterWrapper.PutParameterByName(_smParameterWrapper.TableParameter,
_smParameterWrapper.TableName);
        Console.WriteLine(
            "\nLet's also substitute bad credentials for one of the instances in
the target group so that it can't\n" +
            "access the DynamoDB recommendation table.\n"
        );
        await _autoScalerWrapper.CreateInstanceProfileWithName(
            _autoScalerWrapper.BadCredsPolicyName,
            _autoScalerWrapper.BadCredsRoleName,
```

```
            _autoScalerWrapper.BadCredsProfileName,
            ssmOnlyPolicy,
            new List<string> { "AmazonSSMManagedInstanceCore" }
        );
        var instances = await
_autoScalerWrapper.GetInstancesByGroupName(_autoScalerWrapper.GroupName);
        var badInstanceId = instances.First();
        var instanceProfile = await
_autoScalerWrapper.GetInstanceProfile(badInstanceId);
        Console.WriteLine(
            $"Replacing the profile for instance {badInstanceId} with a profile
that contains\n" +
            "bad credentials...\n"
        );
        await _autoScalerWrapper.ReplaceInstanceProfile(
            badInstanceId,
            _autoScalerWrapper.BadCredsProfileName,
            instanceProfile.AssociationId
        );
        Console.WriteLine(
            "Now, sending a GET request to the load balancer endpoint returns
either a recommendation or a static response,\n" +
            "depending on which instance is selected by the load balancer.\n"
        );
        if (interactive)
            await DemoActionChoices();

        Console.WriteLine("\nLet's implement a deep health check. For this demo,
a deep health check tests whether");
        Console.WriteLine("the web service can access the DynamoDB table that it
depends on for recommendations. Note that");
        Console.WriteLine("the deep health check is only for ELB routing and not
for Auto Scaling instance health.");
        Console.WriteLine("This kind of deep health check is not recommended for
Auto Scaling instance health, because it");
        Console.WriteLine("risks accidental termination of all instances in the
Auto Scaling group when a dependent service fails.");

        Console.WriteLine("\nBy implementing deep health checks, the load
balancer can detect when one of the instances is failing");
        Console.WriteLine("and take that instance out of rotation.");
```

```
        await
_smParameterWrapper.PutParameterByName(_smParameterWrapper.HealthCheckParameter,
"deep");

        Console.WriteLine($"\nNow, checking target health indicates that the
instance with bad credentials ({badInstanceId})");
        Console.WriteLine("is unhealthy. Note that it might take a minute or two
for the load balancer to detect the unhealthy");
        Console.WriteLine("instance. Sending a GET request to the load balancer
endpoint always returns a recommendation, because");
        Console.WriteLine("the load balancer takes unhealthy instances out of its
rotation.");

        if (interactive)
            await DemoActionChoices();

        Console.WriteLine("\nBecause the instances in this demo are controlled by
an auto scaler, the simplest way to fix an unhealthy");
        Console.WriteLine("instance is to terminate it and let the auto scaler
start a new instance to replace it.");

        await _autoScalerWrapper.TryTerminateInstanceById(badInstanceId);

        Console.WriteLine($"\nEven while the instance is terminating and the new
instance is starting, sending a GET");
        Console.WriteLine("request to the web service continues to get a
successful recommendation response because");
        Console.WriteLine("starts and reports as healthy, it is included in the
load balancing rotation.");
        Console.WriteLine("Note that terminating and replacing an instance
typically takes several minutes, during which time you");
        Console.WriteLine("can see the changing health check status until the new
instance is running and healthy.");

        if (interactive)
            await DemoActionChoices();

        Console.WriteLine("\nIf the recommendation service fails now, deep health
checks mean all instances report as unhealthy.");

        await
_smParameterWrapper.PutParameterByName(_smParameterWrapper.TableParameter,
"this-is-not-a-table");
```

```
        Console.WriteLine($"\nWhen all instances are unhealthy, the load balancer
continues to route requests even to");
        Console.WriteLine("unhealthy instances, allowing them to fail open and
return a static response rather than fail");
        Console.WriteLine("closed and report failure to the customer.");

        if (interactive)
            await DemoActionChoices();
        await _smParameterWrapper.Reset();

        Console.WriteLine(new string('-', 80));
        return true;
    }

    /// <summary>
    /// Clean up the resources from the scenario.
    /// </summary>
    /// <param name="interactive">True to ask the user for cleanup.</param>
    /// <returns>Async task.</returns>
    public static async Task<bool> DestroyResources(bool interactive)
    {
        Console.WriteLine(new string('-', 80));
        Console.WriteLine(
            "To keep things tidy and to avoid unwanted charges on your account,
we can clean up all AWS resources\n" +
            "that were created for this demo."
        );

        if (!interactive || GetYesNoResponse("Do you want to clean up all demo
resources? (y/n) "))
        {
            await
_elasticLoadBalancerWrapper.DeleteLoadBalancerByName(_elasticLoadBalancerWrapper.LoadBal
            await
_elasticLoadBalancerWrapper.DeleteTargetGroupByName(_elasticLoadBalancerWrapper.TargetGr
            await
_autoScalerWrapper.TerminateAndDeleteAutoScalingGroupWithName(_autoScalerWrapper.GroupNa
            await
_autoScalerWrapper.DeleteKeyPairByName(_autoScalerWrapper.KeyPairName);
            await
_autoScalerWrapper.DeleteTemplateByName(_autoScalerWrapper.LaunchTemplateName);
            await _autoScalerWrapper.DeleteInstanceProfile(
                _autoScalerWrapper.BadCredsProfileName,
                _autoScalerWrapper.BadCredsRoleName
```

```
            );
            await
 _recommendations.DestroyDatabaseByName(_recommendations.TableName);
        }
        else
        {
            Console.WriteLine(
                "Ok, we'll leave the resources intact.\n" +
                "Don't forget to delete them when you're done with them or you
 might incur unexpected charges."
            );
        }

        Console.WriteLine(new string('-', 80));
        return true;
    }
```

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 ILogger<AutoScalerWrapper> _logger;

    private readonly string _instanceType = "";
    private readonly string _amiParam = "";
    private readonly string _launchTemplateName = "";
    private readonly string _groupName = "";
    private readonly string _instancePolicyName = "";
    private readonly string _instanceRoleName = "";
    private readonly string _instanceProfileName = "";
    private readonly string _badCredsProfileName = "";
    private readonly string _badCredsRoleName = "";
    private readonly string _badCredsPolicyName = "";
    private readonly string _keyPairName = "";
```

```
    public string GroupName => _groupName;
    public string KeyPairName => _keyPairName;
    public string LaunchTemplateName => _launchTemplateName;
    public string InstancePolicyName => _instancePolicyName;
    public string BadCredsProfileName => _badCredsProfileName;
    public string BadCredsRoleName => _badCredsRoleName;
    public string BadCredsPolicyName => _badCredsPolicyName;

    /// <summary>
    /// Constructor for the AutoScalerWrapper.
    /// </summary>
    /// <param name="amazonAutoScaling">The injected AutoScaling client.</param>
    /// <param name="amazonEc2">The injected EC2 client.</param>
    /// <param name="amazonIam">The injected IAM client.</param>
    /// <param name="amazonSsm">The injected SSM client.</param>
    public AutoScalerWrapper(
        IAmazonAutoScaling amazonAutoScaling,
        IAmazonEC2 amazonEc2,
        IAmazonSimpleSystemsManagement amazonSsm,
        IAmazonIdentityManagementService amazonIam,
        IConfiguration configuration,
        ILogger<AutoScalerWrapper> logger)
    {
        _amazonAutoScaling = amazonAutoScaling;
        _amazonEc2 = amazonEc2;
        _amazonSsm = amazonSsm;
        _amazonIam = amazonIam;
        _logger = logger;

        var prefix = configuration["resourcePrefix"];
        _instanceType = configuration["instanceType"];
        _amiParam = configuration["amiParam"];

        _launchTemplateName = prefix + "-template";
        _groupName = prefix + "-group";
        _instancePolicyName = prefix + "-pol";
        _instanceRoleName = prefix + "-role";
        _instanceProfileName = prefix + "-prof";
        _badCredsPolicyName = prefix + "-bc-pol";
        _badCredsRoleName = prefix + "-bc-role";
        _badCredsProfileName = prefix + "-bc-prof";
        _keyPairName = prefix + "-key-pair";
    }
```

```
   /// <summary>
   /// Create a policy, role, and profile that is associated with instances with
a specified name.
   /// An instance's associated profile defines a role that is assumed by the
   /// instance.The role has attached policies that specify the AWS permissions
granted to
   /// clients that run on the instance.
   /// </summary>
   /// <param name="policyName">Name to use for the policy.</param>
   /// <param name="roleName">Name to use for the role.</param>
   /// <param name="profileName">Name to use for the profile.</param>
   /// <param name="ssmOnlyPolicyFile">Path to a policy file for SSM.</param>
   /// <param name="awsManagedPolicies">AWS Managed policies to be attached to
the role.</param>
   /// <returns>The Arn of the profile.</returns>
   public async Task<string> CreateInstanceProfileWithName(
       string policyName,
       string roleName,
       string profileName,
       string ssmOnlyPolicyFile,
       List<string>? awsManagedPolicies = null)
   {

       var assumeRoleDoc = "{" +
                                   "\"Version\": \"2012-10-17\"," +
                                   "\"Statement\": [{" +
                                        "\"Effect\": \"Allow\"," +
                                        "\"Principal\": {" +
                                        "\"Service\": [" +
                                            "\"ec2.amazonaws.com\"" +
                                        "]" +
                                        "}," +
                                   "\"Action\": \"sts:AssumeRole\"" +
                                   "}]" +
                               "}";

       var policyDocument = await File.ReadAllTextAsync(ssmOnlyPolicyFile);

       var policyArn = "";

       try
       {
           var createPolicyResult = await _amazonIam.CreatePolicyAsync(
               new CreatePolicyRequest
```

```
                {
                    PolicyName = policyName,
                    PolicyDocument = policyDocument
                });
            policyArn = createPolicyResult.Policy.Arn;
        }
        catch (EntityAlreadyExistsException)
        {
            // The policy already exists, so we look it up to get the Arn.
            var policiesPaginator = _amazonIam.Paginators.ListPolicies(
                new ListPoliciesRequest()
                {
                    Scope = PolicyScopeType.Local
                });
            // Get the entire list using the paginator.
            await foreach (var policy in policiesPaginator.Policies)
            {
                if (policy.PolicyName.Equals(policyName))
                {
                    policyArn = policy.Arn;
                }
            }

            if (policyArn == null)
            {
                throw new InvalidOperationException("Policy not found");
            }
        }

        try
        {
            await _amazonIam.CreateRoleAsync(new CreateRoleRequest()
            {
                RoleName = roleName,
                AssumeRolePolicyDocument = assumeRoleDoc,
            });
            await _amazonIam.AttachRolePolicyAsync(new AttachRolePolicyRequest()
            {
                RoleName = roleName,
                PolicyArn = policyArn
            });
            if (awsManagedPolicies != null)
            {
                foreach (var awsPolicy in awsManagedPolicies)
```

```
                {
                    await _amazonIam.AttachRolePolicyAsync(new
AttachRolePolicyRequest()
                    {
                        PolicyArn = $"arn:aws:iam::aws:policy/{awsPolicy}",
                        RoleName = roleName
                    });
                }
            }
        }
        catch (EntityAlreadyExistsException)
        {
            Console.WriteLine("Role already exists.");
        }

        string profileArn = "";
        try
        {
            var profileCreateResponse = await
_amazonIam.CreateInstanceProfileAsync(
                new CreateInstanceProfileRequest()
                {
                    InstanceProfileName = profileName
                });
            // Allow time for the profile to be ready.
            profileArn = profileCreateResponse.InstanceProfile.Arn;
            Thread.Sleep(10000);
            await _amazonIam.AddRoleToInstanceProfileAsync(
                new AddRoleToInstanceProfileRequest()
                {
                    InstanceProfileName = profileName,
                    RoleName = roleName
                });

        }
        catch (EntityAlreadyExistsException)
        {
            Console.WriteLine("Policy already exists.");
            var profileGetResponse = await _amazonIam.GetInstanceProfileAsync(
                new GetInstanceProfileRequest()
                {
                    InstanceProfileName = profileName
                });
            profileArn = profileGetResponse.InstanceProfile.Arn;
```

```
        }
        return profileArn;
    }

    /// <summary>
    /// Create a new key pair and save the file.
    /// </summary>
    /// <param name="newKeyPairName">The name of the new key pair.</param>
    /// <returns>Async task.</returns>
    public async Task CreateKeyPair(string newKeyPairName)
    {
        try
        {
            var keyResponse = await _amazonEc2.CreateKeyPairAsync(
                new CreateKeyPairRequest() { KeyName = newKeyPairName });
            await File.WriteAllTextAsync($"{newKeyPairName}.pem",
                keyResponse.KeyPair.KeyMaterial);
            Console.WriteLine($"Created key pair {newKeyPairName}.");
        }
        catch (AlreadyExistsException)
        {
            Console.WriteLine("Key pair already exists.");
        }
    }

    /// <summary>
    /// Delete the key pair and file by name.
    /// </summary>
    /// <param name="deleteKeyPairName">The key pair to delete.</param>
    /// <returns>Async task.</returns>
    public async Task DeleteKeyPairByName(string deleteKeyPairName)
    {
        try
        {
            await _amazonEc2.DeleteKeyPairAsync(
                new DeleteKeyPairRequest() { KeyName = deleteKeyPairName });
            File.Delete($"{deleteKeyPairName}.pem");
        }
        catch (FileNotFoundException)
        {
            Console.WriteLine($"Key pair {deleteKeyPairName} not found.");
        }
    }
```

```
   /// <summary>
   /// Creates an Amazon EC2 launch template to use with Amazon EC2 Auto
Scaling.
   /// The launch template specifies a Bash script in its user data field that
runs after
   /// the instance is started. This script installs the Python packages and
starts a Python
   /// web server on the instance.
   /// </summary>
   /// <param name="startupScriptPath">The path to a Bash script file that is
run.</param>
   /// <param name="instancePolicyPath">The path to a permissions policy to
create and attach to the profile.</param>
   /// <returns>The template object.</returns>
   public async Task<Amazon.EC2.Model.LaunchTemplate> CreateTemplate(string
startupScriptPath, string instancePolicyPath)
   {
       try
       {
           await CreateKeyPair(_keyPairName);
           await CreateInstanceProfileWithName(_instancePolicyName,
_instanceRoleName,
               _instanceProfileName, instancePolicyPath);

           var startServerText = await File.ReadAllTextAsync(startupScriptPath);
           var plainTextBytes =
System.Text.Encoding.UTF8.GetBytes(startServerText);

           var amiLatest = await _amazonSsm.GetParameterAsync(
               new GetParameterRequest() { Name = _amiParam });
           var amiId = amiLatest.Parameter.Value;
           var launchTemplateResponse = await
_amazonEc2.CreateLaunchTemplateAsync(
               new CreateLaunchTemplateRequest()
               {
                   LaunchTemplateName = _launchTemplateName,
                   LaunchTemplateData = new RequestLaunchTemplateData()
                   {
                       InstanceType = _instanceType,
                       ImageId = amiId,
                       IamInstanceProfile =
                           new

LaunchTemplateIamInstanceProfileSpecificationRequest()
```

```
                            {
                                Name = _instanceProfileName
                            },
                        KeyName = _keyPairName,
                        UserData = System.Convert.ToBase64String(plainTextBytes)
                    }
                });
            return launchTemplateResponse.LaunchTemplate;
        }
        catch (AmazonEC2Exception ec2Exception)
        {
            if (ec2Exception.ErrorCode ==
"InvalidLaunchTemplateName.AlreadyExistsException")
            {
                _logger.LogError($"Could not create the template, the name
{_launchTemplateName} already exists. " +
                                 $"Please try again with a unique name.");
            }

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

    /// <summary>
    /// Get a list of Availability Zones in the AWS Region of the Amazon EC2
Client.
    /// </summary>
    /// <returns>A list of availability zones.</returns>
    public async Task<List<string>> DescribeAvailabilityZones()
    {
        try
        {
            var zoneResponse = await _amazonEc2.DescribeAvailabilityZonesAsync(
                new DescribeAvailabilityZonesRequest());
            return zoneResponse.AvailabilityZones.Select(z =>
z.ZoneName).ToList();
        }
        catch (AmazonEC2Exception ec2Exception)
```

```
        {
            _logger.LogError($"An Amazon EC2 error occurred while listing
 availability zones.: {ec2Exception.Message}");
            throw;
        }
        catch (Exception ex)
        {
            _logger.LogError($"An error occurred while listing availability
 zones.: {ex.Message}");
            throw;
        }
    }

    /// <summary>
    /// Create an EC2 Auto Scaling group of a specified size and name.
    /// </summary>
    /// <param name="groupSize">The size for the group.</param>
    /// <param name="groupName">The name for the group.</param>
    /// <param name="availabilityZones">The availability zones for the group.</
param>
    /// <returns>Async task.</returns>
    public async Task CreateGroupOfSize(int groupSize, string groupName,
 List<string> availabilityZones)
    {
        try
        {
            await _amazonAutoScaling.CreateAutoScalingGroupAsync(
                new CreateAutoScalingGroupRequest()
                {
                    AutoScalingGroupName = groupName,
                    AvailabilityZones = availabilityZones,
                    LaunchTemplate =
                        new
 Amazon.AutoScaling.Model.LaunchTemplateSpecification()
                        {
                            LaunchTemplateName = _launchTemplateName,
                            Version = "$Default"
                        },
                    MaxSize = groupSize,
                    MinSize = groupSize
                });
            Console.WriteLine($"Created EC2 Auto Scaling group {groupName} with
 size {groupSize}.");
        }
```

```
        catch (EntityAlreadyExistsException)
        {
            Console.WriteLine($"EC2 Auto Scaling group {groupName} already
exists.");
        }
    }

    /// <summary>
    /// Get the default VPC for the account.
    /// </summary>
    /// <returns>The default VPC object.</returns>
    public async Task<Vpc> GetDefaultVpc()
    {
        try
        {
            var vpcResponse = await _amazonEc2.DescribeVpcsAsync(
                new DescribeVpcsRequest()
                {
                    Filters = new List<Amazon.EC2.Model.Filter>()
                    {
                        new("is-default", new List<string>() { "true" })
                    }
                });
            return vpcResponse.Vpcs[0];
        }
        catch (AmazonEC2Exception ec2Exception)
        {
            if (ec2Exception.ErrorCode == "UnauthorizedOperation")
            {
                _logger.LogError(ec2Exception, $"You do not have the necessary
permissions to describe VPCs.");
            }

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

    /// <summary>
```

```
    /// Get all the subnets for a Vpc in a set of availability zones.
    /// </summary>
    /// <param name="vpcId">The Id of the Vpc.</param>
    /// <param name="availabilityZones">The list of availability zones.</param>
    /// <returns>The collection of subnet objects.</returns>
    public async Task<List<Subnet>> GetAllVpcSubnetsForZones(string vpcId,
List<string> availabilityZones)
    {
        try
        {
            var subnets = new List<Subnet>();
            var subnetPaginator = _amazonEc2.Paginators.DescribeSubnets(
                new DescribeSubnetsRequest()
                {
                    Filters = new List<Amazon.EC2.Model.Filter>()
                    {
                        new("vpc-id", new List<string>() { vpcId }),
                        new("availability-zone", availabilityZones),
                        new("default-for-az", new List<string>() { "true" })
                    }
                });

            // Get the entire list using the paginator.
            await foreach (var subnet in subnetPaginator.Subnets)
            {
                subnets.Add(subnet);
            }

            return subnets;
        }
        catch (AmazonEC2Exception ec2Exception)
        {
            if (ec2Exception.ErrorCode == "InvalidVpcID.NotFound")
            {
                _logger.LogError(ec2Exception, $"The specified VPC ID {vpcId}
does not exist.");
            }

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

```
            throw;
        }
    }

    /// <summary>
    /// Delete a launch template by name.
    /// </summary>
    /// <param name="templateName">The name of the template to delete.</param>
    /// <returns>Async task.</returns>
    public async Task DeleteTemplateByName(string templateName)
    {
        try
        {
            await _amazonEc2.DeleteLaunchTemplateAsync(
                new DeleteLaunchTemplateRequest()
                {
                    LaunchTemplateName = templateName
                });
        }
        catch (AmazonEC2Exception ec2Exception)
        {
            if (ec2Exception.ErrorCode ==
"InvalidLaunchTemplateName.NotFoundException")
            {
                _logger.LogError(
                    $"Could not delete the template, the name
{_launchTemplateName} was not found.");
            }

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

    /// <summary>
    /// Detaches a role from an instance profile, detaches policies from the
role,
    /// and deletes all the resources.
    /// </summary>
```

```
    /// <param name="profileName">The name of the profile to delete.</param>
    /// <param name="roleName">The name of the role to delete.</param>
    /// <returns>Async task.</returns>
    public async Task DeleteInstanceProfile(string profileName, string roleName)
    {
        try
        {
            await _amazonIam.RemoveRoleFromInstanceProfileAsync(
                new RemoveRoleFromInstanceProfileRequest()
                {
                    InstanceProfileName = profileName,
                    RoleName = roleName
                });
            await _amazonIam.DeleteInstanceProfileAsync(
                new DeleteInstanceProfileRequest() { InstanceProfileName =
profileName });
            var attachedPolicies = await
_amazonIam.ListAttachedRolePoliciesAsync(
                new ListAttachedRolePoliciesRequest() { RoleName = roleName });
            foreach (var policy in attachedPolicies.AttachedPolicies)
            {
                await _amazonIam.DetachRolePolicyAsync(
                    new DetachRolePolicyRequest()
                    {
                        RoleName = roleName,
                        PolicyArn = policy.PolicyArn
                    });
                // Delete the custom policies only.
                if (!policy.PolicyArn.StartsWith("arn:aws:iam::aws"))
                {
                    await _amazonIam.DeletePolicyAsync(
                        new Amazon.IdentityManagement.Model.DeletePolicyRequest()
                        {
                            PolicyArn = policy.PolicyArn
                        });
                }
            }

            await _amazonIam.DeleteRoleAsync(
                new DeleteRoleRequest() { RoleName = roleName });
        }
        catch (NoSuchEntityException)
        {
            Console.WriteLine($"Instance profile {profileName} does not exist.");
```

```
        }
    }

    /// <summary>
    /// Gets data about the instances in an EC2 Auto Scaling group by its group
name.
    /// </summary>
    /// <param name="group">The name of the auto scaling group.</param>
    /// <returns>A collection of instance Ids.</returns>
    public async Task<IEnumerable<string>> GetInstancesByGroupName(string group)
    {
        var instanceResponse = await
_amazonAutoScaling.DescribeAutoScalingGroupsAsync(
            new DescribeAutoScalingGroupsRequest()
            {
                AutoScalingGroupNames = new List<string>() { group }
            });
        var instanceIds = instanceResponse.AutoScalingGroups.SelectMany(
            g => g.Instances.Select(i => i.InstanceId));
        return instanceIds;
    }

    /// <summary>
    /// Get the instance profile association data for an instance.
    /// </summary>
    /// <param name="instanceId">The Id of the instance.</param>
    /// <returns>Instance profile associations data.</returns>
    public async Task<IamInstanceProfileAssociation> GetInstanceProfile(string
instanceId)
    {
        try
        {
            var response = await
_amazonEc2.DescribeIamInstanceProfileAssociationsAsync(
                new DescribeIamInstanceProfileAssociationsRequest()
                {
                    Filters = new List<Amazon.EC2.Model.Filter>()
                    {
                        new("instance-id", new List<string>() { instanceId })
                    },
                });
            return response.IamInstanceProfileAssociations[0];
        }
        catch (AmazonEC2Exception ec2Exception)
```

```
        {
            if (ec2Exception.ErrorCode == "InvalidInstanceID.NotFound")
            {
                _logger.LogError(ec2Exception, $"Instance {instanceId} not
found");
            }

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

    /// <summary>
    /// Replace the profile associated with a running instance. After the profile
is replaced, the instance
    /// is rebooted to ensure that it uses the new profile. When the instance is
ready, Systems Manager is
    /// used to restart the Python web server.
    /// </summary>
    /// <param name="instanceId">The Id of the instance to update.</param>
    /// <param name="credsProfileName">The name of the new profile to associate
with the specified instance.</param>
    /// <param name="associationId">The Id of the existing profile association
for the instance.</param>
    /// <returns>Async task.</returns>
    public async Task ReplaceInstanceProfile(string instanceId, string
credsProfileName, string associationId)
    {
        try
        {
            await _amazonEc2.ReplaceIamInstanceProfileAssociationAsync(
                new ReplaceIamInstanceProfileAssociationRequest()
                {
                    AssociationId = associationId,
                    IamInstanceProfile = new IamInstanceProfileSpecification()
                    {
                        Name = credsProfileName
                    }
                });
```

```
            // Allow time before resetting.
            Thread.Sleep(25000);

            await _amazonEc2.RebootInstancesAsync(
                new RebootInstancesRequest(new List<string>() { instanceId }));
            Thread.Sleep(25000);
            var instanceReady = false;
            var retries = 5;
            while (retries-- > 0 && !instanceReady)
            {
                var instancesPaginator =
                    _amazonSsm.Paginators.DescribeInstanceInformation(
                        new DescribeInstanceInformationRequest());
                // Get the entire list using the paginator.
                await foreach (var instance in
instancesPaginator.InstanceInformationList)
                {
                    instanceReady = instance.InstanceId == instanceId;
                    if (instanceReady)
                    {
                        break;
                    }
                }
            }
            Console.WriteLine("Waiting for instance to be running.");
            await WaitForInstanceState(instanceId, InstanceStateName.Running);
            Console.WriteLine("Instance ready.");
            Console.WriteLine($"Sending restart command to instance
{instanceId}");
            await _amazonSsm.SendCommandAsync(
                new SendCommandRequest()
                {
                    InstanceIds = new List<string>() { instanceId },
                    DocumentName = "AWS-RunShellScript",
                    Parameters = new Dictionary<string, List<string>>()
                    {
                        {
                            "commands",
                            new List<string>() { "cd / && sudo python3 server.py
80" }
                        }
                    }
                });
```

```
            Console.WriteLine($"Restarted the web server on instance
{instanceId}");
        }
        catch (AmazonEC2Exception ec2Exception)
        {
            if (ec2Exception.ErrorCode == "InvalidInstanceID.NotFound")
            {
                _logger.LogError(ec2Exception, $"Instance {instanceId} not
found");
            }

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

    /// <summary>
    /// Try to terminate an instance by its Id.
    /// </summary>
    /// <param name="instanceId">The Id of the instance to terminate.</param>
    /// <returns>Async task.</returns>
    public async Task TryTerminateInstanceById(string instanceId)
    {
        var stopping = false;
        Console.WriteLine($"Stopping {instanceId}...");
        while (!stopping)
        {
            try
            {
                await
_amazonAutoScaling.TerminateInstanceInAutoScalingGroupAsync(
                    new TerminateInstanceInAutoScalingGroupRequest()
                    {
                        InstanceId = instanceId,
                        ShouldDecrementDesiredCapacity = false
                    });
                stopping = true;
            }
            catch (ScalingActivityInProgressException)
```

```
            {
                Console.WriteLine($"Scaling activity in progress for
{instanceId}. Waiting...");
                Thread.Sleep(10000);
            }
        }
    }

    /// <summary>
    /// Tries to delete the EC2 Auto Scaling group. If the group is in use or in
progress,
    /// waits and retries until the group is successfully deleted.
    /// </summary>
    /// <param name="groupName">The name of the group to try to delete.</param>
    /// <returns>Async task.</returns>
    public async Task TryDeleteGroupByName(string groupName)
    {
        var stopped = false;
        while (!stopped)
        {
            try
            {
                await _amazonAutoScaling.DeleteAutoScalingGroupAsync(
                    new DeleteAutoScalingGroupRequest()
                    {
                        AutoScalingGroupName = groupName
                    });
                stopped = true;
            }
            catch (Exception e)
                when ((e is ScalingActivityInProgressException)
                      || (e is Amazon.AutoScaling.Model.ResourceInUseException))
            {
                Console.WriteLine($"Some instances are still running.
Waiting...");
                Thread.Sleep(10000);
            }
        }
    }

    /// <summary>
    /// Terminate instances and delete the Auto Scaling group by name.
    /// </summary>
    /// <param name="groupName">The name of the group to delete.</param>
```

```
    /// <returns>Async task.</returns>
    public async Task TerminateAndDeleteAutoScalingGroupWithName(string
groupName)
    {
        var describeGroupsResponse = await
_amazonAutoScaling.DescribeAutoScalingGroupsAsync(
            new DescribeAutoScalingGroupsRequest()
            {
                AutoScalingGroupNames = new List<string>() { groupName }
            });
        if (describeGroupsResponse.AutoScalingGroups.Any())
        {
            // Update the size to 0.
            await _amazonAutoScaling.UpdateAutoScalingGroupAsync(
                new UpdateAutoScalingGroupRequest()
                {
                    AutoScalingGroupName = groupName,
                    MinSize = 0
                });
            var group = describeGroupsResponse.AutoScalingGroups[0];
            foreach (var instance in group.Instances)
            {
                await TryTerminateInstanceById(instance.InstanceId);
            }

            await TryDeleteGroupByName(groupName);
        }
        else
        {
            Console.WriteLine($"No groups found with name {groupName}.");
        }
    }


    /// <summary>
    /// Get the default security group for a specified Vpc.
    /// </summary>
    /// <param name="vpc">The Vpc to search.</param>
    /// <returns>The default security group.</returns>
    public async Task<SecurityGroup> GetDefaultSecurityGroupForVpc(Vpc vpc)
    {
        var groupResponse = await _amazonEc2.DescribeSecurityGroupsAsync(
            new DescribeSecurityGroupsRequest()
            {
```

```
                Filters = new List<Amazon.EC2.Model.Filter>()
                {
                    new ("group-name", new List<string>() { "default" }),
                    new ("vpc-id", new List<string>() { vpc.VpcId })
                }
            });
        return groupResponse.SecurityGroups[0];
    }

    /// <summary>
    /// Verify the default security group of a Vpc allows ingress from the
calling computer.
    /// This can be done by allowing ingress from this computer's IP address.
    /// In some situations, such as connecting from a corporate network, you must
instead specify
    /// a prefix list Id. You can also temporarily open the port to any IP
address while running this example.
    /// If you do, be sure to remove public access when you're done.
    /// </summary>
    /// <param name="vpc">The group to check.</param>
    /// <param name="port">The port to verify.</param>
    /// <param name="ipAddress">This computer's IP address.</param>
    /// <returns>True if the ip address is allowed on the group.</returns>
    public bool VerifyInboundPortForGroup(SecurityGroup group, int port, string
ipAddress)
    {
        var portIsOpen = false;
        foreach (var ipPermission in group.IpPermissions)
        {
            if (ipPermission.FromPort == port)
            {
                foreach (var ipRange in ipPermission.Ipv4Ranges)
                {
                    var cidr = ipRange.CidrIp;
                    if (cidr.StartsWith(ipAddress) || cidr == "0.0.0.0/0")
                    {
                        portIsOpen = true;
                    }
                }

                if (ipPermission.PrefixListIds.Any())
                {
                    portIsOpen = true;
                }
```

```
                if (!portIsOpen)
                {
                    Console.WriteLine("The inbound rule does not appear to be
open to either this computer's IP\n" +
                                      "address, to all IP addresses (0.0.0.0/0),
or to a prefix list ID.");
                }
                else
                {
                    break;
                }
            }
        }

        return portIsOpen;
    }

    /// <summary>
    /// Add an ingress rule to the specified security group that allows access on
the
    /// specified port from the specified IP address.
    /// </summary>
    /// <param name="groupId">The Id of the security group to modify.</param>
    /// <param name="port">The port to open.</param>
    /// <param name="ipAddress">The IP address to allow access.</param>
    /// <returns>Async task.</returns>
    public async Task OpenInboundPort(string groupId, int port, string ipAddress)
    {
        await _amazonEc2.AuthorizeSecurityGroupIngressAsync(
            new AuthorizeSecurityGroupIngressRequest()
            {
                GroupId = groupId,
                IpPermissions = new List<IpPermission>()
                {
                    new IpPermission()
                    {
                        FromPort = port,
                        ToPort = port,
                        IpProtocol = "tcp",
                        Ipv4Ranges = new List<IpRange>()
                        {
                            new IpRange() { CidrIp = $"{ipAddress}/32" }
                        }
```

```
                    }
                }
            });
    }

    /// <summary>
    /// Attaches an Elastic Load Balancing (ELB) target group to this EC2 Auto
 Scaling group.
    /// The
    /// </summary>
    /// <param name="autoScalingGroupName">The name of the Auto Scaling group.</
param>
    /// <param name="targetGroupArn">The Arn for the target group.</param>
    /// <returns>Async task.</returns>
    public async Task AttachLoadBalancerToGroup(string autoScalingGroupName,
 string targetGroupArn)
    {
        await _amazonAutoScaling.AttachLoadBalancerTargetGroupsAsync(
            new AttachLoadBalancerTargetGroupsRequest()
            {
                AutoScalingGroupName = autoScalingGroupName,
                TargetGroupARNs = new List<string>() { targetGroupArn }
            });
    }

    /// <summary>
    /// Wait until an EC2 instance is in a specified state.
    /// </summary>
    /// <param name="instanceId">The instance Id.</param>
    /// <param name="stateName">The state to wait for.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> WaitForInstanceState(string instanceId,
 InstanceStateName stateName)
    {
        var request = new DescribeInstancesRequest
        {
            InstanceIds = new List<string> { instanceId }
        };

        // Wait until the instance is in the specified state.
        var hasState = false;
        do
        {
            // Wait 5 seconds.
```

```
            Thread.Sleep(5000);

            // Check for the desired state.
            var response = await _amazonEc2.DescribeInstancesAsync(request);
            var instance = response.Reservations[0].Instances[0];
            hasState = instance.State.Name == stateName;
            Console.Write(". ");
        } while (!hasState);

        return hasState;
    }
}
```

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.Write($"Creating table {tableName}...");
            var createRequest = new CreateTableRequest()
            {
                TableName = tableName,
                AttributeDefinitions = new List<AttributeDefinition>()
                    {
                        new AttributeDefinition()
                        {
                            AttributeName = "MediaType",
                            AttributeType = ScalarAttributeType.S
                        },
                        new AttributeDefinition()
                        {
                            AttributeName = "ItemId",
                            AttributeType = ScalarAttributeType.N
                        }
                    },
                KeySchema = new List<KeySchemaElement>()
                    {
                        new KeySchemaElement()
                        {
                            AttributeName = "MediaType",
                            KeyType = KeyType.HASH
                        },
                        new KeySchemaElement()
```

```
                        {
                            AttributeName = "ItemId",
                            KeyType = KeyType.RANGE
                        }
                    },
                ProvisionedThroughput = new ProvisionedThroughput()
                {
                    ReadCapacityUnits = 5,
                    WriteCapacityUnits = 5
                }
            };
            await _amazonDynamoDb.CreateTableAsync(createRequest);

            // Wait until the table is ACTIVE and then report success.
            Console.Write("\nWaiting for table to become active...");

            var request = new DescribeTableRequest
            {
                TableName = tableName
            };

            TableStatus status;
            do
            {
                Thread.Sleep(2000);

                var describeTableResponse = await
_amazonDynamoDb.DescribeTableAsync(request);
                status = describeTableResponse.Table.TableStatus;

                Console.Write(".");
            }
            while (status != "ACTIVE");

            return status == TableStatus.ACTIVE;
        }
        catch (ResourceInUseException)
        {
            Console.WriteLine($"Table {tableName} already exists.");
            return false;
        }
    }

    /// <summary>
```

```
    /// Populate the database table with data from a specified path.
    /// </summary>
    /// <param name="databaseTableName">The name of the table.</param>
    /// <param name="recommendationsPath">The path of the recommendations data.</
param>
    /// <returns>Async task.</returns>
    public async Task PopulateDatabase(string databaseTableName, string
 recommendationsPath)
    {
        var recommendationsText = await
 File.ReadAllTextAsync(recommendationsPath);
        var records =

 JsonSerializer.Deserialize<RecommendationModel[]>(recommendationsText);
        var batchWrite = _context.CreateBatchWrite<RecommendationModel>();

        foreach (var record in records!)
        {
            batchWrite.AddPutItem(record);
        }

        await batchWrite.ExecuteAsync();
    }

    /// <summary>
    /// Delete the recommendation table by name.
    /// </summary>
    /// <param name="tableName">The name of the recommendation table.</param>
    /// <returns>Async task.</returns>
    public async Task DestroyDatabaseByName(string tableName)
    {
        try
        {
            await _amazonDynamoDb.DeleteTableAsync(
                new DeleteTableRequest() { TableName = tableName });
            Console.WriteLine($"Table {tableName} was deleted.");
        }
        catch (ResourceNotFoundException)
        {
            Console.WriteLine($"Table {tableName} not found");
        }
    }
}
```

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 para .NET .
  - AttachLoadBalancerTargetGroups
  - CreateAutoScalingGroup
  - CreateInstanceProfile
  - CreateLaunchTemplate
  - CreateListener
  - CreateLoadBalancer
  - CreateTargetGroup
  - DeleteAutoScalingGroup
  - DeleteInstanceProfile
  - DeleteLaunchTemplate
  - DeleteLoadBalancer

- DeleteTargetGroup
- DescribeAutoScalingGroups
- DescribeAvailabilityZones
- DescribeIamInstanceProfileAssociations
- DescribeInstances
- DescribeLoadBalancers
- DescribeSubnets
- DescribeTargetGroups
- DescribeTargetHealth
- DescribeVpcs
- RebootInstances
- ReplaceIamInstanceProfileAssociation
- TerminateInstanceInAutoScalingGroup
- UpdateAutoScalingGroup

Java

SDK 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 wasSuccessul =
loadBalancer.verifyLoadBalancerEndpoint(elbDnsName);
        if (!wasSuccessul) {
            System.out.println("Couldn't connect to the load balancer, verifying
that the port is open...");
            CloseableHttpClient httpClient = HttpClients.createDefault();

            // Create an HTTP GET request to "http://checkip.amazonaws.com"
            HttpGet httpGet = new HttpGet("http://checkip.amazonaws.com");
            try {
                // Execute the request and get the response
                HttpResponse response = httpClient.execute(httpGet);

                // Read the response content.
```

```
                String ipAddress =
IOUtils.toString(response.getEntity().getContent(),
StandardCharsets.UTF_8).trim();

                // Print the public IP address.
                System.out.println("Public IP Address: " + ipAddress);
                GroupInfo groupInfo = autoScaler.verifyInboundPort(vpcId, port,
ipAddress);
                if (!groupInfo.isPortOpen()) {
                    System.out.println("""
                            For this example to work, the default security group
for your default VPC must
                            allow access from this computer. You can either add
it automatically from this
                            example or add it yourself using the AWS Management
Console.
                            """);

                    System.out.println(
                            "Do you want to add a rule to security group " +
groupInfo.getGroupName() + " to allow");
                    System.out.println("inbound traffic on port " + port + " from
your computer's IP address (y/n) ");
                    String ans = in.nextLine();
                    if ("y".equalsIgnoreCase(ans)) {
                        autoScaler.openInboundPort(groupInfo.getGroupName(),
String.valueOf(port), ipAddress);
                        System.out.println("Security group rule added.");
                    } else {
                        System.out.println("No security group rule added.");
                    }
                }

            } catch (AutoScalingException e) {
                e.printStackTrace();
            }
        } else if (wasSuccessul) {
            System.out.println("Your load balancer is ready. You can access it by
browsing to:");
            System.out.println("\t http://" + elbDnsName);
        } else {
            System.out.println("Couldn't get a successful response from the load
balancer endpoint. Troubleshoot by");
```

```
            System.out.println("manually verifying that your VPC and security
group are configured correctly and that");
            System.out.println("you can successfully make a GET request to the
load balancer.");
        }

        System.out.println("Press Enter when you're ready to continue with the
demo.");
        in.nextLine();
    }

    // A method that controls the demo part of the Java program.
    public static void demo(LoadBalancer loadBalancer) throws IOException,
InterruptedException {
        ParameterHelper paramHelper = new ParameterHelper();
        System.out.println("Read the ssm_only_policy.json file");
        String ssmOnlyPolicy = readFileAsString(ssmJSON);

        System.out.println("Resetting parameters to starting values for demo.");
        paramHelper.reset();

        System.out.println(
                """
                        This part of the demonstration shows how to toggle
different parts of the system
                        to create situations where the web service fails, and
shows how using a resilient
                        architecture can keep the web service running in spite
of these failures.

                        At the start, the load balancer endpoint returns
recommendations and reports that all targets are healthy.
                       """);
        demoChoices(loadBalancer);

        System.out.println(
                """
                        The web service running on the EC2 instances gets
recommendations by querying a DynamoDB table.
                        The table name is contained in a Systems Manager
parameter named self.param_helper.table.
                        To simulate a failure of the recommendation service,
let's set this parameter to name a non-existent table.
                       """);
```

```
        paramHelper.put(paramHelper.tableName, "this-is-not-a-table");

        System.out.println(
                """
                        \nNow, sending a GET request to the load balancer
endpoint returns a failure code. But, the service reports as
                        healthy to the load balancer because shallow health
checks don't check for failure of the recommendation service.
                        """);
        demoChoices(loadBalancer);

        System.out.println(
                """
                        Instead of failing when the recommendation service fails,
the web service can return a static response.
                        While this is not a perfect solution, it presents the
customer with a somewhat better experience than failure.
                        """);
        paramHelper.put(paramHelper.failureResponse, "static");

        System.out.println("""
                Now, sending a GET request to the load balancer endpoint returns
a static response.
                The service still reports as healthy because health checks are
still shallow.
                """);
        demoChoices(loadBalancer);

        System.out.println("Let's reinstate the recommendation service.");
        paramHelper.put(paramHelper.tableName, paramHelper.dyntable);

        System.out.println("""
                Let's also substitute bad credentials for one of the instances in
the target group so that it can't
                access the DynamoDB recommendation table. We will get an instance
id value.
                """);

        LaunchTemplateCreator templateCreator = new LaunchTemplateCreator();
        AutoScaler autoScaler = new AutoScaler();

        // Create a new instance profile based on badCredsProfileName.
        templateCreator.createInstanceProfile(policyFile, policyName,
badCredsProfileName, roleName);
```

```
        String badInstanceId = autoScaler.getBadInstance(autoScalingGroupName);
        System.out.println("The bad instance id values used for this demo is " +
badInstanceId);

        String profileAssociationId =
autoScaler.getInstanceProfile(badInstanceId);
        System.out.println("The association Id value is " +
profileAssociationId);
        System.out.println("Replacing the profile for instance " + badInstanceId
                + " with a profile that contains bad credentials");
        autoScaler.replaceInstanceProfile(badInstanceId, badCredsProfileName,
profileAssociationId);

        System.out.println(
                """
                        Now, sending a GET request to the load balancer endpoint
returns either a recommendation or a static response,
                        depending on which instance is selected by the load
balancer.
                        """);

        demoChoices(loadBalancer);

        System.out.println("""
                Let's implement a deep health check. For this demo, a deep health
check tests whether
                the web service can access the DynamoDB table that it depends on
for recommendations. Note that
                the deep health check is only for ELB routing and not for Auto
Scaling instance health.
                This kind of deep health check is not recommended for Auto
Scaling instance health, because it
                risks accidental termination of all instances in the Auto Scaling
group when a dependent service fails.
                """);

        System.out.println("""
                By implementing deep health checks, the load balancer can detect
when one of the instances is failing
                and take that instance out of rotation.
                """);

        paramHelper.put(paramHelper.healthCheck, "deep");
```

```
        System.out.println("""
                Now, checking target health indicates that the instance with bad
credentials
                is unhealthy. Note that it might take a minute or two for the
load balancer to detect the unhealthy
                instance. Sending a GET request to the load balancer endpoint
always returns a recommendation, because
                the load balancer takes unhealthy instances out of its rotation.
                """);

        demoChoices(loadBalancer);

        System.out.println(
                """
                        Because the instances in this demo are controlled by an
auto scaler, the simplest way to fix an unhealthy
                        instance is to terminate it and let the auto scaler start
a new instance to replace it.
                        """);
        autoScaler.terminateInstance(badInstanceId);

        System.out.println("""
                Even while the instance is terminating and the new instance is
starting, sending a GET
                request to the web service continues to get a successful
recommendation response because
                the load balancer routes requests to the healthy instances. After
the replacement instance
                starts and reports as healthy, it is included in the load
balancing rotation.
                Note that terminating and replacing an instance typically takes
several minutes, during which time you
                can see the changing health check status until the new instance
is running and healthy.
                """);

        demoChoices(loadBalancer);
        System.out.println(
                "If the recommendation service fails now, deep health checks mean
all instances report as unhealthy.");
        paramHelper.put(paramHelper.tableName, "this-is-not-a-table");

        demoChoices(loadBalancer);
        paramHelper.reset();
```

```
   }

   public static void demoChoices(LoadBalancer loadBalancer) throws IOException,
InterruptedException {
       String[] actions = {
               "Send a GET request to the load balancer endpoint.",
               "Check the health of load balancer targets.",
               "Go to the next part of the demo."
       };
       Scanner scanner = new Scanner(System.in);

       while (true) {
           System.out.println("-".repeat(88));
           System.out.println("See the current state of the service by selecting
one of the following choices:");
           for (int i = 0; i < actions.length; i++) {
               System.out.println(i + ": " + actions[i]);
           }

           try {
               System.out.print("\nWhich action would you like to take? ");
               int choice = scanner.nextInt();
               System.out.println("-".repeat(88));

               switch (choice) {
                   case 0 -> {
                       System.out.println("Request:\n");
                       System.out.println("GET http://" +
loadBalancer.getEndpoint(lbName));
                       CloseableHttpClient httpClient =
HttpClients.createDefault();

                       // Create an HTTP GET request to the ELB.
                       HttpGet httpGet = new HttpGet("http://" +
loadBalancer.getEndpoint(lbName));

                       // Execute the request and get the response.
                       HttpResponse response = httpClient.execute(httpGet);
                       int statusCode =
response.getStatusLine().getStatusCode();
                       System.out.println("HTTP Status Code: " + statusCode);

                       // Display the JSON response
                       BufferedReader reader = new BufferedReader(
```

```
                                new
InputStreamReader(response.getEntity().getContent()));
                        StringBuilder jsonResponse = new StringBuilder();
                        String line;
                        while ((line = reader.readLine()) != null) {
                            jsonResponse.append(line);
                        }
                        reader.close();

                        // Print the formatted JSON response.
                        System.out.println("Full Response:\n");
                        System.out.println(jsonResponse.toString());

                        // Close the HTTP client.
                        httpClient.close();

                    }
                    case 1 -> {
                        System.out.println("\nChecking the health of load
balancer targets:\n");
                        List<TargetHealthDescription> health =
loadBalancer.checkTargetHealth(targetGroupName);
                        for (TargetHealthDescription target : health) {
                            System.out.printf("\tTarget %s on port %d is %s%n",
target.target().id(),
                                    target.target().port(),
target.targetHealth().stateAsString());
                        }
                        System.out.println("""
                                Note that it can take a minute or two for the
health check to update
                                after changes are made.
                                """);
                    }
                    case 2 -> {
                        System.out.println("\nOkay, let's move on.");
                        System.out.println("-".repeat(88));
                        return; // Exit the method when choice is 2
                    }
                    default -> System.out.println("You must choose a value
between 0-2. Please select again.");
                }

            } catch (java.util.InputMismatchException e) {
```

```
                System.out.println("Invalid input. Please select again.");
                scanner.nextLine(); // Clear the input buffer.
            }
        }
    }

    public static String readFileAsString(String filePath) throws IOException {
        byte[] bytes = Files.readAllBytes(Paths.get(filePath));
        return new String(bytes);
    }
}
```

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.instanceInformationList();
            for (InstanceInformation info : instanceInformationList) {
                if (info.instanceId().equals(instanceId)) {
                    instReady = true;
                    break;
                }
            }
        }

        SendCommandRequest sendCommandRequest = SendCommandRequest.builder()
                .instanceIds(instanceId)
                .documentName("AWS-RunShellScript")
                .parameters(Collections.singletonMap("commands",
                        Collections.singletonList("cd / && sudo python3 server.py
80")))
                .build();

        getSSMClient().sendCommand(sendCommandRequest);
        System.out.println("Restarted the Python web server on instance " +
instanceId + ".");
    }

    public void openInboundPort(String secGroupId, String port, String ipAddress)
{
        AuthorizeSecurityGroupIngressRequest ingressRequest =
AuthorizeSecurityGroupIngressRequest.builder()
                .groupName(secGroupId)
                .cidrIp(ipAddress)
                .fromPort(Integer.parseInt(port))
                .build();

        getEc2Client().authorizeSecurityGroupIngress(ingressRequest);
        System.out.format("Authorized ingress to %s on port %s from %s.",
secGroupId, port, ipAddress);
    }
```

```
   /**
    * Detaches a role from an instance profile, detaches policies from the role,
    * and deletes all the resources.
    */
   public void deleteInstanceProfile(String roleName, String profileName) {
       try {
           software.amazon.awssdk.services.iam.model.GetInstanceProfileRequest
getInstanceProfileRequest =
software.amazon.awssdk.services.iam.model.GetInstanceProfileRequest
                   .builder()
                   .instanceProfileName(profileName)
                   .build();

           GetInstanceProfileResponse response =
getIAMClient().getInstanceProfile(getInstanceProfileRequest);
           String name = response.instanceProfile().instanceProfileName();
           System.out.println(name);

           RemoveRoleFromInstanceProfileRequest profileRequest =
RemoveRoleFromInstanceProfileRequest.builder()
                   .instanceProfileName(profileName)
                   .roleName(roleName)
                   .build();

           getIAMClient().removeRoleFromInstanceProfile(profileRequest);
           DeleteInstanceProfileRequest deleteInstanceProfileRequest =
DeleteInstanceProfileRequest.builder()
                   .instanceProfileName(profileName)
                   .build();

           getIAMClient().deleteInstanceProfile(deleteInstanceProfileRequest);
           System.out.println("Deleted instance profile " + profileName);

           DeleteRoleRequest deleteRoleRequest = DeleteRoleRequest.builder()
                   .roleName(roleName)
                   .build();

           // List attached role policies.
           ListAttachedRolePoliciesResponse rolesResponse = getIAMClient()
                   .listAttachedRolePolicies(role -> role.roleName(roleName));
           List<AttachedPolicy> attachedPolicies =
rolesResponse.attachedPolicies();
           for (AttachedPolicy attachedPolicy : attachedPolicies) {
```

```
                DetachRolePolicyRequest request =
DetachRolePolicyRequest.builder()
                        .roleName(roleName)
                        .policyArn(attachedPolicy.policyArn())
                        .build();

                getIAMClient().detachRolePolicy(request);
                System.out.println("Detached and deleted policy " +
attachedPolicy.policyName());
            }

            getIAMClient().deleteRole(deleteRoleRequest);
            System.out.println("Instance profile and role deleted.");

        } catch (IamException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }

    public void deleteTemplate(String templateName) {
        getEc2Client().deleteLaunchTemplate(name ->
name.launchTemplateName(templateName));
        System.out.format(templateName + " was deleted.");
    }

    public void deleteAutoScaleGroup(String groupName) {
        DeleteAutoScalingGroupRequest deleteAutoScalingGroupRequest =
DeleteAutoScalingGroupRequest.builder()
                .autoScalingGroupName(groupName)
                .forceDelete(true)
                .build();


getAutoScalingClient().deleteAutoScalingGroup(deleteAutoScalingGroupRequest);
        System.out.println(groupName + " was deleted.");
    }

    /*
     * Verify the default security group of the specified VPC allows ingress from
     * this
     * computer. This can be done by allowing ingress from this computer's IP
     * address. In some situations, such as connecting from a corporate network,
you
```

```
    * must instead specify a prefix list ID. You can also temporarily open the
port
    * to
    * any IP address while running this example. If you do, be sure to remove
    * public
    * access when you're done.
    *
    */
   public GroupInfo verifyInboundPort(String VPC, int port, String ipAddress) {
       boolean portIsOpen = false;
       GroupInfo groupInfo = new GroupInfo();
       try {
           Filter filter = Filter.builder()
                   .name("group-name")
                   .values("default")
                   .build();

           Filter filter1 = Filter.builder()
                   .name("vpc-id")
                   .values(VPC)
                   .build();

           DescribeSecurityGroupsRequest securityGroupsRequest =
DescribeSecurityGroupsRequest.builder()
                   .filters(filter, filter1)
                   .build();

           DescribeSecurityGroupsResponse securityGroupsResponse =
getEc2Client()
                   .describeSecurityGroups(securityGroupsRequest);
           String securityGroup =
securityGroupsResponse.securityGroups().get(0).groupName();
           groupInfo.setGroupName(securityGroup);

           for (SecurityGroup secGroup :
securityGroupsResponse.securityGroups()) {
               System.out.println("Found security group: " +
secGroup.groupId());

               for (IpPermission ipPermission : secGroup.ipPermissions()) {
                   if (ipPermission.fromPort() == port) {
                       System.out.println("Found inbound rule: " +
ipPermission);
                       for (IpRange ipRange : ipPermission.ipRanges()) {
```

```
                            String cidrIp = ipRange.cidrIp();
                            if (cidrIp.startsWith(ipAddress) ||
cidrIp.equals("0.0.0.0/0")) {
                                System.out.println(cidrIp + " is applicable");
                                portIsOpen = true;
                            }
                        }

                        if (!ipPermission.prefixListIds().isEmpty()) {
                            System.out.println("Prefix lList is applicable");
                            portIsOpen = true;
                        }

                        if (!portIsOpen) {
                            System.out
                                    .println("The inbound rule does not appear to
be open to either this computer's IP,"
                                            + " all IP addresses (0.0.0.0/0), or
to a prefix list ID.");
                        } else {
                            break;
                        }
                    }
                }
            }

        } catch (AutoScalingException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
        }

        groupInfo.setPortOpen(portIsOpen);
        return groupInfo;
    }

    /*
     * Attaches an Elastic Load Balancing (ELB) target group to this EC2 Auto
     * Scaling group.
     * The target group specifies how the load balancer forward requests to the
     * instances
     * in the group.
     */
    public void attachLoadBalancerTargetGroup(String asGroupName, String
targetGroupARN) {
        try {
```

```
            AttachLoadBalancerTargetGroupsRequest targetGroupsRequest =
AttachLoadBalancerTargetGroupsRequest.builder()
                    .autoScalingGroupName(asGroupName)
                    .targetGroupARNs(targetGroupARN)
                    .build();


getAutoScalingClient().attachLoadBalancerTargetGroups(targetGroupsRequest);
            System.out.println("Attached load balancer to " + asGroupName);

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

    // Creates an EC2 Auto Scaling group with the specified size.
    public String[] createGroup(int groupSize, String templateName, String
autoScalingGroupName) {

        // Get availability zones.

software.amazon.awssdk.services.ec2.model.DescribeAvailabilityZonesRequest
zonesRequest =
software.amazon.awssdk.services.ec2.model.DescribeAvailabilityZonesRequest
                .builder()
                .build();

        DescribeAvailabilityZonesResponse zonesResponse =
getEc2Client().describeAvailabilityZones(zonesRequest);
        List<String> availabilityZoneNames =
zonesResponse.availabilityZones().stream()

.map(software.amazon.awssdk.services.ec2.model.AvailabilityZone::zoneName)
                .collect(Collectors.toList());

        String availabilityZones = String.join(",", availabilityZoneNames);
        LaunchTemplateSpecification specification =
LaunchTemplateSpecification.builder()
                .launchTemplateName(templateName)
                .version("$Default")
                .build();

        String[] zones = availabilityZones.split(",");
```

```
        CreateAutoScalingGroupRequest groupRequest =
CreateAutoScalingGroupRequest.builder()
                .launchTemplate(specification)
                .availabilityZones(zones)
                .maxSize(groupSize)
                .minSize(groupSize)
                .autoScalingGroupName(autoScalingGroupName)
                .build();

        try {
            getAutoScalingClient().createAutoScalingGroup(groupRequest);

        } catch (AutoScalingException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        System.out.println("Created an EC2 Auto Scaling group named " +
autoScalingGroupName);
        return zones;
    }

    public String getDefaultVPC() {
        // Define the filter.
        Filter defaultFilter = Filter.builder()
                .name("is-default")
                .values("true")
                .build();

        software.amazon.awssdk.services.ec2.model.DescribeVpcsRequest request =
software.amazon.awssdk.services.ec2.model.DescribeVpcsRequest
                .builder()
                .filters(defaultFilter)
                .build();

        DescribeVpcsResponse response = getEc2Client().describeVpcs(request);
        return response.vpcs().get(0).vpcId();
    }

    // Gets the default subnets in a VPC for a specified list of Availability
Zones.
    public List<Subnet> getSubnets(String vpcId, String[] availabilityZones) {
        List<Subnet> subnets = null;
        Filter vpcFilter = Filter.builder()
                .name("vpc-id")
```

```
                .values(vpcId)
                .build();

        Filter azFilter = Filter.builder()
                .name("availability-zone")
                .values(availabilityZones)
                .build();

        Filter defaultForAZ = Filter.builder()
                .name("default-for-az")
                .values("true")
                .build();

        DescribeSubnetsRequest request = DescribeSubnetsRequest.builder()
                .filters(vpcFilter, azFilter, defaultForAZ)
                .build();

        DescribeSubnetsResponse response =
getEc2Client().describeSubnets(request);
        subnets = response.subnets();
        return subnets;
    }

    // Gets data about the instances in the EC2 Auto Scaling group.
    public String getBadInstance(String groupName) {
        DescribeAutoScalingGroupsRequest request =
DescribeAutoScalingGroupsRequest.builder()
                .autoScalingGroupNames(groupName)
                .build();

        DescribeAutoScalingGroupsResponse response =
getAutoScalingClient().describeAutoScalingGroups(request);
        AutoScalingGroup autoScalingGroup = response.autoScalingGroups().get(0);
        List<String> instanceIds = autoScalingGroup.instances().stream()
                .map(instance -> instance.instanceId())
                .collect(Collectors.toList());

        String[] instanceIdArray = instanceIds.toArray(new String[0]);
        for (String instanceId : instanceIdArray) {
            System.out.println("Instance ID: " + instanceId);
            return instanceId;
        }
        return "";
    }
```

```
   // Gets data about the profile associated with an instance.
   public String getInstanceProfile(String instanceId) {
       Filter filter = Filter.builder()
               .name("instance-id")
               .values(instanceId)
               .build();

       DescribeIamInstanceProfileAssociationsRequest associationsRequest =
DescribeIamInstanceProfileAssociationsRequest
               .builder()
               .filters(filter)
               .build();

       DescribeIamInstanceProfileAssociationsResponse response = getEc2Client()
               .describeIamInstanceProfileAssociations(associationsRequest);
       return response.iamInstanceProfileAssociations().get(0).associationId();
   }

   public void deleteRolesPolicies(String policyName, String roleName, String
InstanceProfile) {
       ListPoliciesRequest listPoliciesRequest =
ListPoliciesRequest.builder().build();
       ListPoliciesResponse listPoliciesResponse =
getIAMClient().listPolicies(listPoliciesRequest);
       for (Policy policy : listPoliciesResponse.policies()) {
           if (policy.policyName().equals(policyName)) {
               // List the entities (users, groups, roles) that are attached to
the policy.

software.amazon.awssdk.services.iam.model.ListEntitiesForPolicyRequest
listEntitiesRequest =
software.amazon.awssdk.services.iam.model.ListEntitiesForPolicyRequest
                       .builder()
                       .policyArn(policy.arn())
                       .build();
               ListEntitiesForPolicyResponse listEntitiesResponse = iamClient
                       .listEntitiesForPolicy(listEntitiesRequest);
               if (!listEntitiesResponse.policyGroups().isEmpty() || !
listEntitiesResponse.policyUsers().isEmpty()
                       || !listEntitiesResponse.policyRoles().isEmpty()) {
                   // Detach the policy from any entities it is attached to.
                   DetachRolePolicyRequest detachPolicyRequest =
DetachRolePolicyRequest.builder()
```

```
                            .policyArn(policy.arn())
                            .roleName(roleName) // Specify the name of the IAM
role
                            .build();

                    getIAMClient().detachRolePolicy(detachPolicyRequest);
                    System.out.println("Policy detached from entities.");
                }

                // Now, you can delete the policy.
                DeletePolicyRequest deletePolicyRequest =
DeletePolicyRequest.builder()
                        .policyArn(policy.arn())
                        .build();

                getIAMClient().deletePolicy(deletePolicyRequest);
                System.out.println("Policy deleted successfully.");
                break;
            }
        }

        // List the roles associated with the instance profile
        ListInstanceProfilesForRoleRequest listRolesRequest =
ListInstanceProfilesForRoleRequest.builder()
                .roleName(roleName)
                .build();

        // Detach the roles from the instance profile
        ListInstanceProfilesForRoleResponse listRolesResponse =
iamClient.listInstanceProfilesForRole(listRolesRequest);
        for (software.amazon.awssdk.services.iam.model.InstanceProfile profile :
listRolesResponse.instanceProfiles()) {
            RemoveRoleFromInstanceProfileRequest removeRoleRequest =
RemoveRoleFromInstanceProfileRequest.builder()
                    .instanceProfileName(InstanceProfile)
                    .roleName(roleName) // Remove the extra dot here
                    .build();

            getIAMClient().removeRoleFromInstanceProfile(removeRoleRequest);
            System.out.println("Role " + roleName + " removed from instance
profile " + InstanceProfile);
        }

        // Delete the instance profile after removing all roles
```

```
        DeleteInstanceProfileRequest deleteInstanceProfileRequest =
 DeleteInstanceProfileRequest.builder()
                .instanceProfileName(InstanceProfile)
                .build();

        getIAMClient().deleteInstanceProfile(r ->
 r.instanceProfileName(InstanceProfile));
        System.out.println(InstanceProfile + " Deleted");
        System.out.println("All roles and policies are deleted.");
    }
}
```

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)
                    .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
- ReplaceIamInstanceProfileAssociation
- 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 "node:url";

if (process.argv[1] === fileURLToPath(import.meta.url)) {
  parseScenarioArgs(scenarios, {
    name: "Resilient Workflow",
    synopsis:
      "node index.js --scenario <deploy | demo | destroy> [-h|--help] [-y|--yes]
 [-v|--verbose]",
    description: "Deploy and interact with scalable EC2 instances.",
  });
}
```

Cree los pasos para implementar todos los recursos.

```
import { join } from "node:path";
import { readFileSync, writeFileSync } from "node:fs";
```

```
import axios from "axios";

import {
  BatchWriteItemCommand,
  CreateTableCommand,
  DynamoDBClient,
  waitUntilTableExists,
} from "@aws-sdk/client-dynamodb";
import {
  EC2Client,
  CreateKeyPairCommand,
  CreateLaunchTemplateCommand,
  DescribeAvailabilityZonesCommand,
  DescribeVpcsCommand,
  DescribeSubnetsCommand,
  DescribeSecurityGroupsCommand,
  AuthorizeSecurityGroupIngressCommand,
} from "@aws-sdk/client-ec2";
import {
  IAMClient,
  CreatePolicyCommand,
  CreateRoleCommand,
  CreateInstanceProfileCommand,
  AddRoleToInstanceProfileCommand,
  AttachRolePolicyCommand,
  waitUntilInstanceProfileExists,
} from "@aws-sdk/client-iam";
import { SSMClient, GetParameterCommand } from "@aws-sdk/client-ssm";
import {
  CreateAutoScalingGroupCommand,
  AutoScalingClient,
  AttachLoadBalancerTargetGroupsCommand,
} from "@aws-sdk/client-auto-scaling";
import {
  CreateListenerCommand,
  CreateLoadBalancerCommand,
  CreateTargetGroupCommand,
  ElasticLoadBalancingV2Client,
  waitUntilLoadBalancerAvailable,
} from "@aws-sdk/client-elastic-load-balancing-v2";

import {
  ScenarioOutput,
  ScenarioInput,
```

```
  ScenarioAction,
} from "@aws-doc-sdk-examples/lib/scenario/index.js";
import { saveState } from "@aws-doc-sdk-examples/lib/scenario/steps-common.js";
import { retry } from "@aws-doc-sdk-examples/lib/utils/util-timers.js";

import { MESSAGES, NAMES, RESOURCES_PATH, ROOT } from "./constants.js";
import { initParamsSteps } from "./steps-reset-params.js";

/**
 * @type {import('@aws-doc-sdk-examples/lib/scenario.js').Step[]}
 */
export const deploySteps = [
  new ScenarioOutput("introduction", MESSAGES.introduction, { header: true }),
  new ScenarioInput("confirmDeployment", MESSAGES.confirmDeployment, {
    type: "confirm",
  }),
  new ScenarioAction(
    "handleConfirmDeployment",
    (c) => c.confirmDeployment === false && process.exit(),
  ),
  new ScenarioOutput(
    "creatingTable",
    MESSAGES.creatingTable.replace("${TABLE_NAME}", NAMES.tableName),
  ),
  new ScenarioAction("createTable", async () => {
    const client = new DynamoDBClient({});
    await client.send(
      new CreateTableCommand({
        TableName: NAMES.tableName,
        ProvisionedThroughput: {
          ReadCapacityUnits: 5,
          WriteCapacityUnits: 5,
        },
        AttributeDefinitions: [
          {
            AttributeName: "MediaType",
            AttributeType: "S",
          },
          {
            AttributeName: "ItemId",
            AttributeType: "N",
          },
        ],
        KeySchema: [
```

```
          {
            AttributeName: "MediaType",
            KeyType: "HASH",
          },
          {
            AttributeName: "ItemId",
            KeyType: "RANGE",
          },
        ],
      }),
    );
    await waitUntilTableExists({ client }, { TableName: NAMES.tableName });
  }),
  new ScenarioOutput(
    "createdTable",
    MESSAGES.createdTable.replace("${TABLE_NAME}", NAMES.tableName),
  ),
  new ScenarioOutput(
    "populatingTable",
    MESSAGES.populatingTable.replace("${TABLE_NAME}", NAMES.tableName),
  ),
  new ScenarioAction("populateTable", () => {
    const client = new DynamoDBClient({});
    /**
     * @type {{ default: import("@aws-sdk/client-dynamodb").PutRequest['Item']
[] }}
     */
    const recommendations = JSON.parse(
      readFileSync(join(RESOURCES_PATH, "recommendations.json")),
    );

    return client.send(
      new BatchWriteItemCommand({
        RequestItems: {
          [NAMES.tableName]: recommendations.map((item) => ({
            PutRequest: { Item: item },
          })),
        },
      }),
    );
  }),
  new ScenarioOutput(
    "populatedTable",
    MESSAGES.populatedTable.replace("${TABLE_NAME}", NAMES.tableName),
```

```
  ),
  new ScenarioOutput(
    "creatingKeyPair",
    MESSAGES.creatingKeyPair.replace("${KEY_PAIR_NAME}", NAMES.keyPairName),
  ),
  new ScenarioAction("createKeyPair", async () => {
    const client = new EC2Client({});
    const { KeyMaterial } = await client.send(
      new CreateKeyPairCommand({
        KeyName: NAMES.keyPairName,
      }),
    );

    writeFileSync(`${NAMES.keyPairName}.pem`, KeyMaterial, { mode: 0o600 });
  }),
  new ScenarioOutput(
    "createdKeyPair",
    MESSAGES.createdKeyPair.replace("${KEY_PAIR_NAME}", NAMES.keyPairName),
  ),
  new ScenarioOutput(
    "creatingInstancePolicy",
    MESSAGES.creatingInstancePolicy.replace(
      "${INSTANCE_POLICY_NAME}",
      NAMES.instancePolicyName,
    ),
  ),
  new ScenarioAction("createInstancePolicy", async (state) => {
    const client = new IAMClient({});
    const {
      Policy: { Arn },
    } = await client.send(
      new CreatePolicyCommand({
        PolicyName: NAMES.instancePolicyName,
        PolicyDocument: readFileSync(
          join(RESOURCES_PATH, "instance_policy.json"),
        ),
      }),
    );
    state.instancePolicyArn = Arn;
  }),
  new ScenarioOutput("createdInstancePolicy", (state) =>
    MESSAGES.createdInstancePolicy
      .replace("${INSTANCE_POLICY_NAME}", NAMES.instancePolicyName)
      .replace("${INSTANCE_POLICY_ARN}", state.instancePolicyArn),
```

```
  ),
  new ScenarioOutput(
    "creatingInstanceRole",
    MESSAGES.creatingInstanceRole.replace(
      "${INSTANCE_ROLE_NAME}",
      NAMES.instanceRoleName,
    ),
  ),
  new ScenarioAction("createInstanceRole", () => {
    const client = new IAMClient({});
    return client.send(
      new CreateRoleCommand({
        RoleName: NAMES.instanceRoleName,
        AssumeRolePolicyDocument: readFileSync(
          join(ROOT, "assume-role-policy.json"),
        ),
      }),
    );
  }),
  new ScenarioOutput(
    "createdInstanceRole",
    MESSAGES.createdInstanceRole.replace(
      "${INSTANCE_ROLE_NAME}",
      NAMES.instanceRoleName,
    ),
  ),
  new ScenarioOutput(
    "attachingPolicyToRole",
    MESSAGES.attachingPolicyToRole
      .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName)
      .replace("${INSTANCE_POLICY_NAME}", NAMES.instancePolicyName),
  ),
  new ScenarioAction("attachPolicyToRole", async (state) => {
    const client = new IAMClient({});
    await client.send(
      new AttachRolePolicyCommand({
        RoleName: NAMES.instanceRoleName,
        PolicyArn: state.instancePolicyArn,
      }),
    );
  }),
  new ScenarioOutput(
    "attachedPolicyToRole",
    MESSAGES.attachedPolicyToRole
```

```
      .replace("${INSTANCE_POLICY_NAME}", NAMES.instancePolicyName)
      .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName),
  ),
  new ScenarioOutput(
    "creatingInstanceProfile",
    MESSAGES.creatingInstanceProfile.replace(
      "${INSTANCE_PROFILE_NAME}",
      NAMES.instanceProfileName,
    ),
  ),
  new ScenarioAction("createInstanceProfile", async (state) => {
    const client = new IAMClient({});
    const {
      InstanceProfile: { Arn },
    } = await client.send(
      new CreateInstanceProfileCommand({
        InstanceProfileName: NAMES.instanceProfileName,
      }),
    );
    state.instanceProfileArn = Arn;

    await waitUntilInstanceProfileExists(
      { client },
      { InstanceProfileName: NAMES.instanceProfileName },
    );
  }),
  new ScenarioOutput("createdInstanceProfile", (state) =>
    MESSAGES.createdInstanceProfile
      .replace("${INSTANCE_PROFILE_NAME}", NAMES.instanceProfileName)
      .replace("${INSTANCE_PROFILE_ARN}", state.instanceProfileArn),
  ),
  new ScenarioOutput(
    "addingRoleToInstanceProfile",
    MESSAGES.addingRoleToInstanceProfile
      .replace("${INSTANCE_PROFILE_NAME}", NAMES.instanceProfileName)
      .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName),
  ),
  new ScenarioAction("addRoleToInstanceProfile", () => {
    const client = new IAMClient({});
    return client.send(
      new AddRoleToInstanceProfileCommand({
        RoleName: NAMES.instanceRoleName,
        InstanceProfileName: NAMES.instanceProfileName,
      }),
```

```
    );
  }),
  new ScenarioOutput(
    "addedRoleToInstanceProfile",
    MESSAGES.addedRoleToInstanceProfile
      .replace("${INSTANCE_PROFILE_NAME}", NAMES.instanceProfileName)
      .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName),
  ),
  ...initParamsSteps,
  new ScenarioOutput("creatingLaunchTemplate", MESSAGES.creatingLaunchTemplate),
  new ScenarioAction("createLaunchTemplate", async () => {
    const ssmClient = new SSMClient({});
    const { Parameter } = await ssmClient.send(
      new GetParameterCommand({
        Name: "/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2",
      }),
    );
    const ec2Client = new EC2Client({});
    await ec2Client.send(
      new CreateLaunchTemplateCommand({
        LaunchTemplateName: NAMES.launchTemplateName,
        LaunchTemplateData: {
          InstanceType: "t3.micro",
          ImageId: Parameter.Value,
          IamInstanceProfile: { Name: NAMES.instanceProfileName },
          UserData: readFileSync(
            join(RESOURCES_PATH, "server_startup_script.sh"),
          ).toString("base64"),
          KeyName: NAMES.keyPairName,
        },
      }),
    );
  }),
  new ScenarioOutput(
    "createdLaunchTemplate",
    MESSAGES.createdLaunchTemplate.replace(
      "${LAUNCH_TEMPLATE_NAME}",
      NAMES.launchTemplateName,
    ),
  ),
  new ScenarioOutput(
    "creatingAutoScalingGroup",
    MESSAGES.creatingAutoScalingGroup.replace(
      "${AUTO_SCALING_GROUP_NAME}",
```

```
      NAMES.autoScalingGroupName,
    ),
  ),
  new ScenarioAction("createAutoScalingGroup", async (state) => {
    const ec2Client = new EC2Client({});
    const { AvailabilityZones } = await ec2Client.send(
      new DescribeAvailabilityZonesCommand({}),
    );
    state.availabilityZoneNames = AvailabilityZones.map((az) => az.ZoneName);
    const autoScalingClient = new AutoScalingClient({});
    await retry({ intervalInMs: 1000, maxRetries: 30 }, () =>
      autoScalingClient.send(
        new CreateAutoScalingGroupCommand({
          AvailabilityZones: state.availabilityZoneNames,
          AutoScalingGroupName: NAMES.autoScalingGroupName,
          LaunchTemplate: {
            LaunchTemplateName: NAMES.launchTemplateName,
            Version: "$Default",
          },
          MinSize: 3,
          MaxSize: 3,
        }),
      ),
    );
  }),
  new ScenarioOutput(
    "createdAutoScalingGroup",
    /**
     * @param {{ availabilityZoneNames: string[] }} state
     */
    (state) =>
      MESSAGES.createdAutoScalingGroup
        .replace("${AUTO_SCALING_GROUP_NAME}", NAMES.autoScalingGroupName)
        .replace(
          "${AVAILABILITY_ZONE_NAMES}",
          state.availabilityZoneNames.join(", "),
        ),
  ),
  new ScenarioInput("confirmContinue", MESSAGES.confirmContinue, {
    type: "confirm",
  }),
  new ScenarioOutput("loadBalancer", MESSAGES.loadBalancer),
  new ScenarioOutput("gettingVpc", MESSAGES.gettingVpc),
  new ScenarioAction("getVpc", async (state) => {
```

```
    const client = new EC2Client({});
    const { Vpcs } = await client.send(
      new DescribeVpcsCommand({
        Filters: [{ Name: "is-default", Values: ["true"] }],
      }),
    );
    state.defaultVpc = Vpcs[0].VpcId;
  }),
  new ScenarioOutput("gotVpc", (state) =>
    MESSAGES.gotVpc.replace("${VPC_ID}", state.defaultVpc),
  ),
  new ScenarioOutput("gettingSubnets", MESSAGES.gettingSubnets),
  new ScenarioAction("getSubnets", async (state) => {
    const client = new EC2Client({});
    const { Subnets } = await client.send(
      new DescribeSubnetsCommand({
        Filters: [
          { Name: "vpc-id", Values: [state.defaultVpc] },
          { Name: "availability-zone", Values: state.availabilityZoneNames },
          { Name: "default-for-az", Values: ["true"] },
        ],
      }),
    );
    state.subnets = Subnets.map((subnet) => subnet.SubnetId);
  }),
  new ScenarioOutput(
    "gotSubnets",
    /**
     * @param {{ subnets: string[] }} state
     */
    (state) =>
      MESSAGES.gotSubnets.replace("${SUBNETS}", state.subnets.join(", ")),
  ),
  new ScenarioOutput(
    "creatingLoadBalancerTargetGroup",
    MESSAGES.creatingLoadBalancerTargetGroup.replace(
      "${TARGET_GROUP_NAME}",
      NAMES.loadBalancerTargetGroupName,
    ),
  ),
  new ScenarioAction("createLoadBalancerTargetGroup", async (state) => {
    const client = new ElasticLoadBalancingV2Client({});
    const { TargetGroups } = await client.send(
      new CreateTargetGroupCommand({
```

```
        Name: NAMES.loadBalancerTargetGroupName,
        Protocol: "HTTP",
        Port: 80,
        HealthCheckPath: "/healthcheck",
        HealthCheckIntervalSeconds: 10,
        HealthCheckTimeoutSeconds: 5,
        HealthyThresholdCount: 2,
        UnhealthyThresholdCount: 2,
        VpcId: state.defaultVpc,
      }),
    );
    const targetGroup = TargetGroups[0];
    state.targetGroupArn = targetGroup.TargetGroupArn;
    state.targetGroupProtocol = targetGroup.Protocol;
    state.targetGroupPort = targetGroup.Port;
  }),
  new ScenarioOutput(
    "createdLoadBalancerTargetGroup",
    MESSAGES.createdLoadBalancerTargetGroup.replace(
      "${TARGET_GROUP_NAME}",
      NAMES.loadBalancerTargetGroupName,
    ),
  ),
  new ScenarioOutput(
    "creatingLoadBalancer",
    MESSAGES.creatingLoadBalancer.replace("${LB_NAME}", NAMES.loadBalancerName),
  ),
  new ScenarioAction("createLoadBalancer", async (state) => {
    const client = new ElasticLoadBalancingV2Client({});
    const { LoadBalancers } = await client.send(
      new CreateLoadBalancerCommand({
        Name: NAMES.loadBalancerName,
        Subnets: state.subnets,
      }),
    );
    state.loadBalancerDns = LoadBalancers[0].DNSName;
    state.loadBalancerArn = LoadBalancers[0].LoadBalancerArn;
    await waitUntilLoadBalancerAvailable(
      { client },
      { Names: [NAMES.loadBalancerName] },
    );
  }),
  new ScenarioOutput("createdLoadBalancer", (state) =>
    MESSAGES.createdLoadBalancer
```

```
      .replace("${LB_NAME}", NAMES.loadBalancerName)
      .replace("${DNS_NAME}", state.loadBalancerDns),
  ),
  new ScenarioOutput(
    "creatingListener",
    MESSAGES.creatingLoadBalancerListener
      .replace("${LB_NAME}", NAMES.loadBalancerName)
      .replace("${TARGET_GROUP_NAME}", NAMES.loadBalancerTargetGroupName),
  ),
  new ScenarioAction("createListener", async (state) => {
    const client = new ElasticLoadBalancingV2Client({});
    const { Listeners } = await client.send(
      new CreateListenerCommand({
        LoadBalancerArn: state.loadBalancerArn,
        Protocol: state.targetGroupProtocol,
        Port: state.targetGroupPort,
        DefaultActions: [
          { Type: "forward", TargetGroupArn: state.targetGroupArn },
        ],
      }),
    );
    const listener = Listeners[0];
    state.loadBalancerListenerArn = listener.ListenerArn;
  }),
  new ScenarioOutput("createdListener", (state) =>
    MESSAGES.createdLoadBalancerListener.replace(
      "${LB_LISTENER_ARN}",
      state.loadBalancerListenerArn,
    ),
  ),
  new ScenarioOutput(
    "attachingLoadBalancerTargetGroup",
    MESSAGES.attachingLoadBalancerTargetGroup
      .replace("${TARGET_GROUP_NAME}", NAMES.loadBalancerTargetGroupName)
      .replace("${AUTO_SCALING_GROUP_NAME}", NAMES.autoScalingGroupName),
  ),
  new ScenarioAction("attachLoadBalancerTargetGroup", async (state) => {
    const client = new AutoScalingClient({});
    await client.send(
      new AttachLoadBalancerTargetGroupsCommand({
        AutoScalingGroupName: NAMES.autoScalingGroupName,
        TargetGroupARNs: [state.targetGroupArn],
      }),
    );
```

```
  }),
  new ScenarioOutput(
    "attachedLoadBalancerTargetGroup",
    MESSAGES.attachedLoadBalancerTargetGroup,
  ),
  new ScenarioOutput("verifyingInboundPort", MESSAGES.verifyingInboundPort),
  new ScenarioAction(
    "verifyInboundPort",
    /**
     *
     * @param {{ defaultSecurityGroup: import('@aws-sdk/client-
ec2').SecurityGroup}} state
     */
    async (state) => {
      const client = new EC2Client({});
      const { SecurityGroups } = await client.send(
        new DescribeSecurityGroupsCommand({
          Filters: [{ Name: "group-name", Values: ["default"] }],
        }),
      );
      if (!SecurityGroups) {
        state.verifyInboundPortError = new Error(MESSAGES.noSecurityGroups);
      }
      state.defaultSecurityGroup = SecurityGroups[0];

      /**
       * @type {string}
       */
      const ipResponse = (await axios.get("http://checkip.amazonaws.com")).data;
      state.myIp = ipResponse.trim();
      const myIpRules = state.defaultSecurityGroup.IpPermissions.filter(
        ({ IpRanges }) =>
          IpRanges.some(
            ({ CidrIp }) =>
              CidrIp.startsWith(state.myIp) || CidrIp === "0.0.0.0/0",
          ),
      )
        .filter(({ IpProtocol }) => IpProtocol === "tcp")
        .filter(({ FromPort }) => FromPort === 80);

      state.myIpRules = myIpRules;
    },
  ),
  new ScenarioOutput(
```

```
    "verifiedInboundPort",
    /**
     * @param {{ myIpRules: any[] }} state
     */
    (state) => {
      if (state.myIpRules.length > 0) {
        return MESSAGES.foundIpRules.replace(
          "${IP_RULES}",
          JSON.stringify(state.myIpRules, null, 2),
        );
      }
      return MESSAGES.noIpRules;
    },
  ),
  new ScenarioInput(
    "shouldAddInboundRule",
    /**
     * @param {{ myIpRules: any[] }} state
     */
    (state) => {
      if (state.myIpRules.length > 0) {
        return false;
      }
      return MESSAGES.noIpRules;
    },
    { type: "confirm" },
  ),
  new ScenarioAction(
    "addInboundRule",
    /**
     * @param {{ defaultSecurityGroup: import('@aws-sdk/client-
ec2').SecurityGroup }} state
     */
    async (state) => {
      if (!state.shouldAddInboundRule) {
        return;
      }

      const client = new EC2Client({});
      await client.send(
        new AuthorizeSecurityGroupIngressCommand({
          GroupId: state.defaultSecurityGroup.GroupId,
          CidrIp: `${state.myIp}/32`,
          FromPort: 80,
```

```
          ToPort: 80,
          IpProtocol: "tcp",
        }),
      );
    },
  ),
  new ScenarioOutput("addedInboundRule", (state) => {
    if (state.shouldAddInboundRule) {
      return MESSAGES.addedInboundRule.replace("${IP_ADDRESS}", state.myIp);
    }
    return false;
  }),
  new ScenarioOutput("verifyingEndpoint", (state) =>
    MESSAGES.verifyingEndpoint.replace("${DNS_NAME}", state.loadBalancerDns),
  ),
  new ScenarioAction("verifyEndpoint", async (state) => {
    try {
      const response = await retry({ intervalInMs: 2000, maxRetries: 30 }, () =>
        axios.get(`http://${state.loadBalancerDns}`),
      );
      state.endpointResponse = JSON.stringify(response.data, null, 2);
    } catch (e) {
      state.verifyEndpointError = e;
    }
  }),
  new ScenarioOutput("verifiedEndpoint", (state) => {
    if (state.verifyEndpointError) {
      console.error(state.verifyEndpointError);
    } else {
      return MESSAGES.verifiedEndpoint.replace(
        "${ENDPOINT_RESPONSE}",
        state.endpointResponse,
      );
    }
  }),
  saveState,
];
```

Cree los pasos para ejecutar la demostración.

```
import { readFileSync } from "node:fs";
import { join } from "node:path";
```

```
import axios from "axios";

import {
  DescribeTargetGroupsCommand,
  DescribeTargetHealthCommand,
  ElasticLoadBalancingV2Client,
} from "@aws-sdk/client-elastic-load-balancing-v2";
import {
  DescribeInstanceInformationCommand,
  PutParameterCommand,
  SSMClient,
  SendCommandCommand,
} from "@aws-sdk/client-ssm";
import {
  IAMClient,
  CreatePolicyCommand,
  CreateRoleCommand,
  AttachRolePolicyCommand,
  CreateInstanceProfileCommand,
  AddRoleToInstanceProfileCommand,
  waitUntilInstanceProfileExists,
} from "@aws-sdk/client-iam";
import {
  AutoScalingClient,
  DescribeAutoScalingGroupsCommand,
  TerminateInstanceInAutoScalingGroupCommand,
} from "@aws-sdk/client-auto-scaling";
import {
  DescribeIamInstanceProfileAssociationsCommand,
  EC2Client,
  RebootInstancesCommand,
  ReplaceIamInstanceProfileAssociationCommand,
} from "@aws-sdk/client-ec2";

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

import { MESSAGES, NAMES, RESOURCES_PATH } from "./constants.js";
import { findLoadBalancer } from "./shared.js";
```

```
const getRecommendation = new ScenarioAction(
  "getRecommendation",
  async (state) => {
    const loadBalancer = await findLoadBalancer(NAMES.loadBalancerName);
    if (loadBalancer) {
      state.loadBalancerDnsName = loadBalancer.DNSName;
      try {
        state.recommendation = (
          await axios.get(`http://${state.loadBalancerDnsName}`)
        ).data;
      } catch (e) {
        state.recommendation = e instanceof Error ? e.message : e;
      }
    } else {
      throw new Error(MESSAGES.demoFindLoadBalancerError);
    }
  },
);

const getRecommendationResult = new ScenarioOutput(
  "getRecommendationResult",
  (state) =>
    `Recommendation:\n${JSON.stringify(state.recommendation, null, 2)}`,
  { preformatted: true },
);

const getHealthCheck = new ScenarioAction("getHealthCheck", async (state) => {
  const client = new ElasticLoadBalancingV2Client({});
  const { TargetGroups } = await client.send(
    new DescribeTargetGroupsCommand({
      Names: [NAMES.loadBalancerTargetGroupName],
    }),
  );

  const { TargetHealthDescriptions } = await client.send(
    new DescribeTargetHealthCommand({
      TargetGroupArn: TargetGroups[0].TargetGroupArn,
    }),
  );
  state.targetHealthDescriptions = TargetHealthDescriptions;
});

const getHealthCheckResult = new ScenarioOutput(
```

```
  "getHealthCheckResult",
  /**
   * @param {{ targetHealthDescriptions: import('@aws-sdk/client-elastic-load-
balancing-v2').TargetHealthDescription[]}} state
   */
  (state) => {
    const status = state.targetHealthDescriptions
      .map((th) => `${th.Target.Id}: ${th.TargetHealth.State}`)
      .join("\n");
    return `Health check:\n${status}`;
  },
  { preformatted: true },
);

const loadBalancerLoop = new ScenarioAction(
  "loadBalancerLoop",
  getRecommendation.action,
  {
    whileConfig: {
      whileFn: ({ loadBalancerCheck }) => loadBalancerCheck,
      input: new ScenarioInput(
        "loadBalancerCheck",
        MESSAGES.demoLoadBalancerCheck,
        {
          type: "confirm",
        },
      ),
      output: getRecommendationResult,
    },
  },
);

const healthCheckLoop = new ScenarioAction(
  "healthCheckLoop",
  getHealthCheck.action,
  {
    whileConfig: {
      whileFn: ({ healthCheck }) => healthCheck,
      input: new ScenarioInput("healthCheck", MESSAGES.demoHealthCheck, {
        type: "confirm",
      }),
      output: getHealthCheckResult,
    },
  },
```

```
);

const statusSteps = [
  getRecommendation,
  getRecommendationResult,
  getHealthCheck,
  getHealthCheckResult,
];

/**
 * @type {import('@aws-doc-sdk-examples/lib/scenario.js').Step[]}
 */
export const demoSteps = [
  new ScenarioOutput("header", MESSAGES.demoHeader, { header: true }),
  new ScenarioOutput("sanityCheck", MESSAGES.demoSanityCheck),
  ...statusSteps,
  new ScenarioInput(
    "brokenDependencyConfirmation",
    MESSAGES.demoBrokenDependencyConfirmation,
    { type: "confirm" },
  ),
  new ScenarioAction("brokenDependency", async (state) => {
    if (!state.brokenDependencyConfirmation) {
      process.exit();
    } else {
      const client = new SSMClient({});
      state.badTableName = `fake-table-${Date.now()}`;
      await client.send(
        new PutParameterCommand({
          Name: NAMES.ssmTableNameKey,
          Value: state.badTableName,
          Overwrite: true,
          Type: "String",
        }),
      );
    }
  }),
  new ScenarioOutput("testBrokenDependency", (state) =>
    MESSAGES.demoTestBrokenDependency.replace(
      "${TABLE_NAME}",
      state.badTableName,
    ),
  ),
  ...statusSteps,
```

```
  new ScenarioInput(
    "staticResponseConfirmation",
    MESSAGES.demoStaticResponseConfirmation,
    { type: "confirm" },
  ),
  new ScenarioAction("staticResponse", async (state) => {
    if (!state.staticResponseConfirmation) {
      process.exit();
    } else {
      const client = new SSMClient({});
      await client.send(
        new PutParameterCommand({
          Name: NAMES.ssmFailureResponseKey,
          Value: "static",
          Overwrite: true,
          Type: "String",
        }),
      );
    }
  }),
  new ScenarioOutput("testStaticResponse", MESSAGES.demoTestStaticResponse),
  ...statusSteps,
  new ScenarioInput(
    "badCredentialsConfirmation",
    MESSAGES.demoBadCredentialsConfirmation,
    { type: "confirm" },
  ),
  new ScenarioAction("badCredentialsExit", (state) => {
    if (!state.badCredentialsConfirmation) {
      process.exit();
    }
  }),
  new ScenarioAction("fixDynamoDBName", async () => {
    const client = new SSMClient({});
    await client.send(
      new PutParameterCommand({
        Name: NAMES.ssmTableNameKey,
        Value: NAMES.tableName,
        Overwrite: true,
        Type: "String",
      }),
    );
  }),
  new ScenarioAction(
```

```
    "badCredentials",
    /**
     * @param {{ targetInstance: import('@aws-sdk/client-auto-
scaling').Instance }} state
     */
    async (state) => {
      await createSsmOnlyInstanceProfile();
      const autoScalingClient = new AutoScalingClient({});
      const { AutoScalingGroups } = await autoScalingClient.send(
        new DescribeAutoScalingGroupsCommand({
          AutoScalingGroupNames: [NAMES.autoScalingGroupName],
        }),
      );
      state.targetInstance = AutoScalingGroups[0].Instances[0];
      const ec2Client = new EC2Client({});
      const { IamInstanceProfileAssociations } = await ec2Client.send(
        new DescribeIamInstanceProfileAssociationsCommand({
          Filters: [
            { Name: "instance-id", Values: [state.targetInstance.InstanceId] },
          ],
        }),
      );
      state.instanceProfileAssociationId =
        IamInstanceProfileAssociations[0].AssociationId;
      await retry({ intervalInMs: 1000, maxRetries: 30 }, () =>
        ec2Client.send(
          new ReplaceIamInstanceProfileAssociationCommand({
            AssociationId: state.instanceProfileAssociationId,
            IamInstanceProfile: { Name: NAMES.ssmOnlyInstanceProfileName },
          }),
        ),
      );

      await ec2Client.send(
        new RebootInstancesCommand({
          InstanceIds: [state.targetInstance.InstanceId],
        }),
      );

      const ssmClient = new SSMClient({});
      await retry({ intervalInMs: 20000, maxRetries: 15 }, async () => {
        const { InstanceInformationList } = await ssmClient.send(
          new DescribeInstanceInformationCommand({}),
        );
```

```
        const instance = InstanceInformationList.find(
          (info) => info.InstanceId === state.targetInstance.InstanceId,
        );

        if (!instance) {
          throw new Error("Instance not found.");
        }
      });

      await ssmClient.send(
        new SendCommandCommand({
          InstanceIds: [state.targetInstance.InstanceId],
          DocumentName: "AWS-RunShellScript",
          Parameters: { commands: ["cd / && sudo python3 server.py 80"] },
        }),
      );
    },
  ),
  new ScenarioOutput(
    "testBadCredentials",
    /**
     * @param {{ targetInstance: import('@aws-sdk/client-
ssm').InstanceInformation}} state
     */
    (state) =>
      MESSAGES.demoTestBadCredentials.replace(
        "${INSTANCE_ID}",
        state.targetInstance.InstanceId,
      ),
  ),
  loadBalancerLoop,
  new ScenarioInput(
    "deepHealthCheckConfirmation",
    MESSAGES.demoDeepHealthCheckConfirmation,
    { type: "confirm" },
  ),
  new ScenarioAction("deepHealthCheckExit", (state) => {
    if (!state.deepHealthCheckConfirmation) {
      process.exit();
    }
  }),
  new ScenarioAction("deepHealthCheck", async () => {
    const client = new SSMClient({});
```

```
    await client.send(
      new PutParameterCommand({
        Name: NAMES.ssmHealthCheckKey,
        Value: "deep",
        Overwrite: true,
        Type: "String",
      }),
    );
  }),
  new ScenarioOutput("testDeepHealthCheck", MESSAGES.demoTestDeepHealthCheck),
  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",
```

```
    }),
  );
  const { InstanceProfile } = await iamClient.send(
    new CreateInstanceProfileCommand({
      InstanceProfileName: NAMES.ssmOnlyInstanceProfileName,
    }),
  );
  await waitUntilInstanceProfileExists(
    { client: iamClient },
    { InstanceProfileName: NAMES.ssmOnlyInstanceProfileName },
  );
  await iamClient.send(
    new AddRoleToInstanceProfileCommand({
      InstanceProfileName: NAMES.ssmOnlyInstanceProfileName,
      RoleName: NAMES.ssmOnlyRoleName,
    }),
  );

  return InstanceProfile;
}
```

Cree los pasos para destruir todos los recursos.

```
import { unlinkSync } from "node:fs";

import { DynamoDBClient, DeleteTableCommand } from "@aws-sdk/client-dynamodb";
import {
  EC2Client,
  DeleteKeyPairCommand,
  DeleteLaunchTemplateCommand,
  RevokeSecurityGroupIngressCommand,
} from "@aws-sdk/client-ec2";
import {
  IAMClient,
  DeleteInstanceProfileCommand,
  RemoveRoleFromInstanceProfileCommand,
  DeletePolicyCommand,
  DeleteRoleCommand,
  DetachRolePolicyCommand,
  paginateListPolicies,
} from "@aws-sdk/client-iam";
import {
```

```
  AutoScalingClient,
  DeleteAutoScalingGroupCommand,
  TerminateInstanceInAutoScalingGroupCommand,
  UpdateAutoScalingGroupCommand,
  paginateDescribeAutoScalingGroups,
} from "@aws-sdk/client-auto-scaling";
import {
  DeleteLoadBalancerCommand,
  DeleteTargetGroupCommand,
  DescribeTargetGroupsCommand,
  ElasticLoadBalancingV2Client,
} from "@aws-sdk/client-elastic-load-balancing-v2";

import {
  ScenarioOutput,
  ScenarioInput,
  ScenarioAction,
} from "@aws-doc-sdk-examples/lib/scenario/index.js";
import { loadState } from "@aws-doc-sdk-examples/lib/scenario/steps-common.js";
import { retry } from "@aws-doc-sdk-examples/lib/utils/util-timers.js";

import { MESSAGES, NAMES } from "./constants.js";
import { findLoadBalancer } from "./shared.js";

/**
 * @type {import('@aws-doc-sdk-examples/lib/scenario.js').Step[]}
 */
export const destroySteps = [
  loadState,
  new ScenarioInput("destroy", MESSAGES.destroy, { type: "confirm" }),
  new ScenarioAction(
    "abort",
    (state) => state.destroy === false && process.exit(),
  ),
  new ScenarioAction("deleteTable", async (c) => {
    try {
      const client = new DynamoDBClient({});
      await client.send(new DeleteTableCommand({ TableName: NAMES.tableName }));
    } catch (e) {
      c.deleteTableError = e;
    }
  }),
  new ScenarioOutput("deleteTableResult", (state) => {
    if (state.deleteTableError) {
```

```
      console.error(state.deleteTableError);
      return MESSAGES.deleteTableError.replace(
        "${TABLE_NAME}",
        NAMES.tableName,
      );
    }
    return MESSAGES.deletedTable.replace("${TABLE_NAME}", NAMES.tableName);
  }),
  new ScenarioAction("deleteKeyPair", async (state) => {
    try {
      const client = new EC2Client({});
      await client.send(
        new DeleteKeyPairCommand({ KeyName: NAMES.keyPairName }),
      );
      unlinkSync(`${NAMES.keyPairName}.pem`);
    } catch (e) {
      state.deleteKeyPairError = e;
    }
  }),
  new ScenarioOutput("deleteKeyPairResult", (state) => {
    if (state.deleteKeyPairError) {
      console.error(state.deleteKeyPairError);
      return MESSAGES.deleteKeyPairError.replace(
        "${KEY_PAIR_NAME}",
        NAMES.keyPairName,
      );
    }
    return MESSAGES.deletedKeyPair.replace(
      "${KEY_PAIR_NAME}",
      NAMES.keyPairName,
    );
  }),
  new ScenarioAction("detachPolicyFromRole", async (state) => {
    try {
      const client = new IAMClient({});
      const policy = await findPolicy(NAMES.instancePolicyName);

      if (!policy) {
        state.detachPolicyFromRoleError = new Error(
          `Policy ${NAMES.instancePolicyName} not found.`,
        );
      } else {
        await client.send(
          new DetachRolePolicyCommand({
```

```
            RoleName: NAMES.instanceRoleName,
            PolicyArn: policy.Arn,
          }),
        );
      }
    } catch (e) {
      state.detachPolicyFromRoleError = e;
    }
  }),
  new ScenarioOutput("detachedPolicyFromRole", (state) => {
    if (state.detachPolicyFromRoleError) {
      console.error(state.detachPolicyFromRoleError);
      return MESSAGES.detachPolicyFromRoleError
        .replace("${INSTANCE_POLICY_NAME}", NAMES.instancePolicyName)
        .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName);
    }
    return MESSAGES.detachedPolicyFromRole
      .replace("${INSTANCE_POLICY_NAME}", NAMES.instancePolicyName)
      .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName);
  }),
  new ScenarioAction("deleteInstancePolicy", async (state) => {
    const client = new IAMClient({});
    const policy = await findPolicy(NAMES.instancePolicyName);

    if (!policy) {
      state.deletePolicyError = new Error(
        `Policy ${NAMES.instancePolicyName} not found.`,
      );
    } else {
      return client.send(
        new DeletePolicyCommand({
          PolicyArn: policy.Arn,
        }),
      );
    }
  }),
  new ScenarioOutput("deletePolicyResult", (state) => {
    if (state.deletePolicyError) {
      console.error(state.deletePolicyError);
      return MESSAGES.deletePolicyError.replace(
        "${INSTANCE_POLICY_NAME}",
        NAMES.instancePolicyName,
      );
    }
```

```
    return MESSAGES.deletedPolicy.replace(
      "${INSTANCE_POLICY_NAME}",
      NAMES.instancePolicyName,
    );
  }),
  new ScenarioAction("removeRoleFromInstanceProfile", async (state) => {
    try {
      const client = new IAMClient({});
      await client.send(
        new RemoveRoleFromInstanceProfileCommand({
          RoleName: NAMES.instanceRoleName,
          InstanceProfileName: NAMES.instanceProfileName,
        }),
      );
    } catch (e) {
      state.removeRoleFromInstanceProfileError = e;
    }
  }),
  new ScenarioOutput("removeRoleFromInstanceProfileResult", (state) => {
    if (state.removeRoleFromInstanceProfile) {
      console.error(state.removeRoleFromInstanceProfileError);
      return MESSAGES.removeRoleFromInstanceProfileError
        .replace("${INSTANCE_PROFILE_NAME}", NAMES.instanceProfileName)
        .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName);
    }
    return MESSAGES.removedRoleFromInstanceProfile
      .replace("${INSTANCE_PROFILE_NAME}", NAMES.instanceProfileName)
      .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName);
  }),
  new ScenarioAction("deleteInstanceRole", async (state) => {
    try {
      const client = new IAMClient({});
      await client.send(
        new DeleteRoleCommand({
          RoleName: NAMES.instanceRoleName,
        }),
      );
    } catch (e) {
      state.deleteInstanceRoleError = e;
    }
  }),
  new ScenarioOutput("deleteInstanceRoleResult", (state) => {
    if (state.deleteInstanceRoleError) {
      console.error(state.deleteInstanceRoleError);
```

```
      return MESSAGES.deleteInstanceRoleError.replace(
        "${INSTANCE_ROLE_NAME}",
        NAMES.instanceRoleName,
      );
    }
    return MESSAGES.deletedInstanceRole.replace(
      "${INSTANCE_ROLE_NAME}",
      NAMES.instanceRoleName,
    );
  }),
  new ScenarioAction("deleteInstanceProfile", async (state) => {
    try {
      const client = new IAMClient({});
      await client.send(
        new DeleteInstanceProfileCommand({
          InstanceProfileName: NAMES.instanceProfileName,
        }),
      );
    } catch (e) {
      state.deleteInstanceProfileError = e;
    }
  }),
  new ScenarioOutput("deleteInstanceProfileResult", (state) => {
    if (state.deleteInstanceProfileError) {
      console.error(state.deleteInstanceProfileError);
      return MESSAGES.deleteInstanceProfileError.replace(
        "${INSTANCE_PROFILE_NAME}",
        NAMES.instanceProfileName,
      );
    }
    return MESSAGES.deletedInstanceProfile.replace(
      "${INSTANCE_PROFILE_NAME}",
      NAMES.instanceProfileName,
    );
  }),
  new ScenarioAction("deleteAutoScalingGroup", async (state) => {
    try {
      await terminateGroupInstances(NAMES.autoScalingGroupName);
      await retry({ intervalInMs: 60000, maxRetries: 60 }, async () => {
        await deleteAutoScalingGroup(NAMES.autoScalingGroupName);
      });
    } catch (e) {
      state.deleteAutoScalingGroupError = e;
    }
```

```
  }),
  new ScenarioOutput("deleteAutoScalingGroupResult", (state) => {
    if (state.deleteAutoScalingGroupError) {
      console.error(state.deleteAutoScalingGroupError);
      return MESSAGES.deleteAutoScalingGroupError.replace(
        "${AUTO_SCALING_GROUP_NAME}",
        NAMES.autoScalingGroupName,
      );
    }
    return MESSAGES.deletedAutoScalingGroup.replace(
      "${AUTO_SCALING_GROUP_NAME}",
      NAMES.autoScalingGroupName,
    );
  }),
  new ScenarioAction("deleteLaunchTemplate", async (state) => {
    const client = new EC2Client({});
    try {
      await client.send(
        new DeleteLaunchTemplateCommand({
          LaunchTemplateName: NAMES.launchTemplateName,
        }),
      );
    } catch (e) {
      state.deleteLaunchTemplateError = e;
    }
  }),
  new ScenarioOutput("deleteLaunchTemplateResult", (state) => {
    if (state.deleteLaunchTemplateError) {
      console.error(state.deleteLaunchTemplateError);
      return MESSAGES.deleteLaunchTemplateError.replace(
        "${LAUNCH_TEMPLATE_NAME}",
        NAMES.launchTemplateName,
      );
    }
    return MESSAGES.deletedLaunchTemplate.replace(
      "${LAUNCH_TEMPLATE_NAME}",
      NAMES.launchTemplateName,
    );
  }),
  new ScenarioAction("deleteLoadBalancer", async (state) => {
    try {
      const client = new ElasticLoadBalancingV2Client({});
      const loadBalancer = await findLoadBalancer(NAMES.loadBalancerName);
      await client.send(
```

```
        new DeleteLoadBalancerCommand({
          LoadBalancerArn: loadBalancer.LoadBalancerArn,
        }),
      );
      await retry({ intervalInMs: 1000, maxRetries: 60 }, async () => {
        const lb = await findLoadBalancer(NAMES.loadBalancerName);
        if (lb) {
          throw new Error("Load balancer still exists.");
        }
      });
    } catch (e) {
      state.deleteLoadBalancerError = e;
    }
  }),
  new ScenarioOutput("deleteLoadBalancerResult", (state) => {
    if (state.deleteLoadBalancerError) {
      console.error(state.deleteLoadBalancerError);
      return MESSAGES.deleteLoadBalancerError.replace(
        "${LB_NAME}",
        NAMES.loadBalancerName,
      );
    }
    return MESSAGES.deletedLoadBalancer.replace(
      "${LB_NAME}",
      NAMES.loadBalancerName,
    );
  }),
  new ScenarioAction("deleteLoadBalancerTargetGroup", async (state) => {
    const client = new ElasticLoadBalancingV2Client({});
    try {
      const { TargetGroups } = await client.send(
        new DescribeTargetGroupsCommand({
          Names: [NAMES.loadBalancerTargetGroupName],
        }),
      );

      await retry({ intervalInMs: 1000, maxRetries: 30 }, () =>
        client.send(
          new DeleteTargetGroupCommand({
            TargetGroupArn: TargetGroups[0].TargetGroupArn,
          }),
        ),
      );
    } catch (e) {
```

```
      state.deleteLoadBalancerTargetGroupError = e;
    }
  }),
  new ScenarioOutput("deleteLoadBalancerTargetGroupResult", (state) => {
    if (state.deleteLoadBalancerTargetGroupError) {
      console.error(state.deleteLoadBalancerTargetGroupError);
      return MESSAGES.deleteLoadBalancerTargetGroupError.replace(
        "${TARGET_GROUP_NAME}",
        NAMES.loadBalancerTargetGroupName,
      );
    }
    return MESSAGES.deletedLoadBalancerTargetGroup.replace(
      "${TARGET_GROUP_NAME}",
      NAMES.loadBalancerTargetGroupName,
    );
  }),
  new ScenarioAction("detachSsmOnlyRoleFromProfile", async (state) => {
    try {
      const client = new IAMClient({});
      await client.send(
        new RemoveRoleFromInstanceProfileCommand({
          InstanceProfileName: NAMES.ssmOnlyInstanceProfileName,
          RoleName: NAMES.ssmOnlyRoleName,
        }),
      );
    } catch (e) {
      state.detachSsmOnlyRoleFromProfileError = e;
    }
  }),
  new ScenarioOutput("detachSsmOnlyRoleFromProfileResult", (state) => {
    if (state.detachSsmOnlyRoleFromProfileError) {
      console.error(state.detachSsmOnlyRoleFromProfileError);
      return MESSAGES.detachSsmOnlyRoleFromProfileError
        .replace("${ROLE_NAME}", NAMES.ssmOnlyRoleName)
        .replace("${PROFILE_NAME}", NAMES.ssmOnlyInstanceProfileName);
    }
    return MESSAGES.detachedSsmOnlyRoleFromProfile
      .replace("${ROLE_NAME}", NAMES.ssmOnlyRoleName)
      .replace("${PROFILE_NAME}", NAMES.ssmOnlyInstanceProfileName);
  }),
  new ScenarioAction("detachSsmOnlyCustomRolePolicy", async (state) => {
    try {
      const iamClient = new IAMClient({});
      const ssmOnlyPolicy = await findPolicy(NAMES.ssmOnlyPolicyName);
```

```
      await iamClient.send(
        new DetachRolePolicyCommand({
          RoleName: NAMES.ssmOnlyRoleName,
          PolicyArn: ssmOnlyPolicy.Arn,
        }),
      );
    } catch (e) {
      state.detachSsmOnlyCustomRolePolicyError = e;
    }
  }),
  new ScenarioOutput("detachSsmOnlyCustomRolePolicyResult", (state) => {
    if (state.detachSsmOnlyCustomRolePolicyError) {
      console.error(state.detachSsmOnlyCustomRolePolicyError);
      return MESSAGES.detachSsmOnlyCustomRolePolicyError
        .replace("${ROLE_NAME}", NAMES.ssmOnlyRoleName)
        .replace("${POLICY_NAME}", NAMES.ssmOnlyPolicyName);
    }
    return MESSAGES.detachedSsmOnlyCustomRolePolicy
      .replace("${ROLE_NAME}", NAMES.ssmOnlyRoleName)
      .replace("${POLICY_NAME}", NAMES.ssmOnlyPolicyName);
  }),
  new ScenarioAction("detachSsmOnlyAWSRolePolicy", async (state) => {
    try {
      const iamClient = new IAMClient({});
      await iamClient.send(
        new DetachRolePolicyCommand({
          RoleName: NAMES.ssmOnlyRoleName,
          PolicyArn: "arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore",
        }),
      );
    } catch (e) {
      state.detachSsmOnlyAWSRolePolicyError = e;
    }
  }),
  new ScenarioOutput("detachSsmOnlyAWSRolePolicyResult", (state) => {
    if (state.detachSsmOnlyAWSRolePolicyError) {
      console.error(state.detachSsmOnlyAWSRolePolicyError);
      return MESSAGES.detachSsmOnlyAWSRolePolicyError
        .replace("${ROLE_NAME}", NAMES.ssmOnlyRoleName)
        .replace("${POLICY_NAME}", "AmazonSSMManagedInstanceCore");
    }
    return MESSAGES.detachedSsmOnlyAWSRolePolicy
      .replace("${ROLE_NAME}", NAMES.ssmOnlyRoleName)
      .replace("${POLICY_NAME}", "AmazonSSMManagedInstanceCore");
```

```
  }),
  new ScenarioAction("deleteSsmOnlyInstanceProfile", async (state) => {
    try {
      const iamClient = new IAMClient({});
      await iamClient.send(
        new DeleteInstanceProfileCommand({
          InstanceProfileName: NAMES.ssmOnlyInstanceProfileName,
        }),
      );
    } catch (e) {
      state.deleteSsmOnlyInstanceProfileError = e;
    }
  }),
  new ScenarioOutput("deleteSsmOnlyInstanceProfileResult", (state) => {
    if (state.deleteSsmOnlyInstanceProfileError) {
      console.error(state.deleteSsmOnlyInstanceProfileError);
      return MESSAGES.deleteSsmOnlyInstanceProfileError.replace(
        "${INSTANCE_PROFILE_NAME}",
        NAMES.ssmOnlyInstanceProfileName,
      );
    }
    return MESSAGES.deletedSsmOnlyInstanceProfile.replace(
      "${INSTANCE_PROFILE_NAME}",
      NAMES.ssmOnlyInstanceProfileName,
    );
  }),
  new ScenarioAction("deleteSsmOnlyPolicy", async (state) => {
    try {
      const iamClient = new IAMClient({});
      const ssmOnlyPolicy = await findPolicy(NAMES.ssmOnlyPolicyName);
      await iamClient.send(
        new DeletePolicyCommand({
          PolicyArn: ssmOnlyPolicy.Arn,
        }),
      );
    } catch (e) {
      state.deleteSsmOnlyPolicyError = e;
    }
  }),
  new ScenarioOutput("deleteSsmOnlyPolicyResult", (state) => {
    if (state.deleteSsmOnlyPolicyError) {
      console.error(state.deleteSsmOnlyPolicyError);
      return MESSAGES.deleteSsmOnlyPolicyError.replace(
        "${POLICY_NAME}",
```

```
        NAMES.ssmOnlyPolicyName,
      );
    }
    return MESSAGES.deletedSsmOnlyPolicy.replace(
      "${POLICY_NAME}",
      NAMES.ssmOnlyPolicyName,
    );
  }),
  new ScenarioAction("deleteSsmOnlyRole", async (state) => {
    try {
      const iamClient = new IAMClient({});
      await iamClient.send(
        new DeleteRoleCommand({
          RoleName: NAMES.ssmOnlyRoleName,
        }),
      );
    } catch (e) {
      state.deleteSsmOnlyRoleError = e;
    }
  }),
  new ScenarioOutput("deleteSsmOnlyRoleResult", (state) => {
    if (state.deleteSsmOnlyRoleError) {
      console.error(state.deleteSsmOnlyRoleError);
      return MESSAGES.deleteSsmOnlyRoleError.replace(
        "${ROLE_NAME}",
        NAMES.ssmOnlyRoleName,
      );
    }
    return MESSAGES.deletedSsmOnlyRole.replace(
      "${ROLE_NAME}",
      NAMES.ssmOnlyRoleName,
    );
  }),
  new ScenarioAction(
    "revokeSecurityGroupIngress",
    async (
      /** @type {{ myIp: string, defaultSecurityGroup: { GroupId: string } }} */
state,
    ) => {
      const ec2Client = new EC2Client({});

      try {
        await ec2Client.send(
          new RevokeSecurityGroupIngressCommand({
```

```
            GroupId: state.defaultSecurityGroup.GroupId,
            CidrIp: `${state.myIp}/32`,
            FromPort: 80,
            ToPort: 80,
            IpProtocol: "tcp",
          }),
        );
      } catch (e) {
        state.revokeSecurityGroupIngressError = e;
      }
    },
  ),
  new ScenarioOutput("revokeSecurityGroupIngressResult", (state) => {
    if (state.revokeSecurityGroupIngressError) {
      console.error(state.revokeSecurityGroupIngressError);
      return MESSAGES.revokeSecurityGroupIngressError.replace(
        "${IP}",
        state.myIp,
      );
    }
    return MESSAGES.revokedSecurityGroupIngress.replace("${IP}", state.myIp);
  }),
];

/**
 * @param {string} policyName
 */
async function findPolicy(policyName) {
  const client = new IAMClient({});
  const paginatedPolicies = paginateListPolicies({ client }, {});
  for await (const page of paginatedPolicies) {
    const policy = page.Policies.find((p) => p.PolicyName === policyName);
    if (policy) {
      return policy;
    }
  }
}

/**
 * @param {string} groupName
 */
async function deleteAutoScalingGroup(groupName) {
  const client = new AutoScalingClient({});
  try {
```

```
    await client.send(
      new DeleteAutoScalingGroupCommand({
        AutoScalingGroupName: groupName,
      }),
    );
  } catch (err) {
    if (!(err instanceof Error)) {
      throw err;
    }
    console.log(err.name);
    throw err;
  }
}

/**
 * @param {string} groupName
 */
async function terminateGroupInstances(groupName) {
  const autoScalingClient = new AutoScalingClient({});
  const group = await findAutoScalingGroup(groupName);
  await autoScalingClient.send(
    new UpdateAutoScalingGroupCommand({
      AutoScalingGroupName: group.AutoScalingGroupName,
      MinSize: 0,
    }),
  );
  for (const i of group.Instances) {
    await retry({ intervalInMs: 1000, maxRetries: 30 }, () =>
      autoScalingClient.send(
        new TerminateInstanceInAutoScalingGroupCommand({
          InstanceId: i.InstanceId,
          ShouldDecrementDesiredCapacity: true,
        }),
      ),
    );
  }
}

async function findAutoScalingGroup(groupName) {
  const client = new AutoScalingClient({});
  const paginatedGroups = paginateDescribeAutoScalingGroups({ client }, {});
  for await (const page of paginatedGroups) {
    const group = page.AutoScalingGroups.find(
      (g) => g.AutoScalingGroupName === groupName,
```

```
    );
    if (group) {
      return group;
    }
  }
  throw new Error(`Auto scaling group ${groupName} not found.`);
}
```

- Para obtener detalles sobre la API, consulte los siguientes temas en la Referencia de la API de AWS SDK para JavaScript .
  - AttachLoadBalancerTargetGroups
  - CreateAutoScalingGroup
  - CreateInstanceProfile
  - CreateLaunchTemplate
  - CreateListener
  - CreateLoadBalancer
  - CreateTargetGroup
  - DeleteAutoScalingGroup
  - DeleteInstanceProfile
  - DeleteLaunchTemplate
  - DeleteLoadBalancer
  - DeleteTargetGroup
  - DescribeAutoScalingGroups
  - DescribeAvailabilityZones
  - DescribeIamInstanceProfileAssociations
  - DescribeInstances
  - DescribeLoadBalancers
  - DescribeSubnets
  - DescribeTargetGroups
  - DescribeTargetHealth
  - DescribeVpcs
  - RebootInstances

- ReplaceIamInstanceProfileAssociation
- TerminateInstanceInAutoScalingGroup
- UpdateAutoScalingGroup

Python

SDK 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:
    """
    Manages the deployment, demonstration, and destruction of resources for the
 resilient service.
    """

    def __init__(
        self,
        resource_path: str,
        recommendation: RecommendationService,
        autoscaler: AutoScalingWrapper,
        loadbalancer: ElasticLoadBalancerWrapper,
        param_helper: ParameterHelper,
    ):
        """
        Initializes the Runner class with the necessary parameters.

        :param resource_path: The path to resource files used by this example,
 such as IAM policies and instance scripts.
        :param recommendation: An instance of the RecommendationService class.
        :param autoscaler: An instance of the AutoScaler class.
        :param loadbalancer: An instance of the LoadBalancer class.
        :param param_helper: An instance of the ParameterHelper class.
        """
        self.resource_path = resource_path
```

```
        self.recommendation = recommendation
        self.autoscaler = autoscaler
        self.loadbalancer = loadbalancer
        self.param_helper = param_helper
        self.protocol = "HTTP"
        self.port = 80
        self.ssh_port = 22

        prefix = "doc-example-resilience"
        self.target_group_name = f"{prefix}-tg"
        self.load_balancer_name = f"{prefix}-lb"

    def deploy(self) -> None:
        """
        Deploys the resources required for the resilient service, including the
DynamoDB table,
        EC2 instances, Auto Scaling group, and load balancer.
        """
        recommendations_path = f"{self.resource_path}/recommendations.json"
        startup_script = f"{self.resource_path}/server_startup_script.sh"
        instance_policy = f"{self.resource_path}/instance_policy.json"

        logging.info("Starting deployment of resources for the resilient
service.")

        logging.info(
            "Creating and populating DynamoDB table '%s'.",
            self.recommendation.table_name,
        )
        self.recommendation.create()
        self.recommendation.populate(recommendations_path)

        logging.info(
            "Creating an EC2 launch template with the startup script '%s'.",
            startup_script,
        )
        self.autoscaler.create_template(startup_script, instance_policy)

        logging.info(
            "Creating an EC2 Auto Scaling group across multiple Availability
Zones."
        )
        zones = self.autoscaler.create_autoscaling_group(3)
```

```
        logging.info("Creating variables that control the flow of the demo.")
        self.param_helper.reset()

        logging.info("Creating Elastic Load Balancing target group and load
 balancer.")

        vpc = self.autoscaler.get_default_vpc()
        subnets = self.autoscaler.get_subnets(vpc["VpcId"], zones)
        target_group = self.loadbalancer.create_target_group(
            self.target_group_name, self.protocol, self.port, vpc["VpcId"]
        )
        self.loadbalancer.create_load_balancer(
            self.load_balancer_name, [subnet["SubnetId"] for subnet in subnets]
        )
        self.loadbalancer.create_listener(self.load_balancer_name, target_group)

        self.autoscaler.attach_load_balancer_target_group(target_group)

        logging.info("Verifying access to the load balancer endpoint.")
        endpoint = self.loadbalancer.get_endpoint(self.load_balancer_name)
        lb_success = self.loadbalancer.verify_load_balancer_endpoint(endpoint)
        current_ip_address = requests.get("http://
checkip.amazonaws.com").text.strip()

        if not lb_success:
            logging.warning(
                "Couldn't connect to the load balancer. Verifying that the port
 is open..."
            )
            sec_group, port_is_open = self.autoscaler.verify_inbound_port(
                vpc, self.port, current_ip_address
            )
            sec_group, ssh_port_is_open = self.autoscaler.verify_inbound_port(
                vpc, self.ssh_port, current_ip_address
            )
            if not port_is_open:
                logging.warning(
                    "The default security group for your VPC must allow access
 from this computer."
                )
                if q.ask(
                    f"Do you want to add a rule to security group
 {sec_group['GroupId']} to allow\n"
```

```
                    f"inbound traffic on port {self.port} from your computer's IP
address of {current_ip_address}? (y/n) ",
                    q.is_yesno,
                ):
                    self.autoscaler.open_inbound_port(
                        sec_group["GroupId"], self.port, current_ip_address
                    )
            if not ssh_port_is_open:
                if q.ask(
                    f"Do you want to add a rule to security group
{sec_group['GroupId']} to allow\n"
                    f"inbound SSH traffic on port {self.ssh_port} for debugging
from your computer's IP address of {current_ip_address}? (y/n) ",
                    q.is_yesno,
                ):
                    self.autoscaler.open_inbound_port(
                        sec_group["GroupId"], self.ssh_port, current_ip_address
                    )
            lb_success =
self.loadbalancer.verify_load_balancer_endpoint(endpoint)

        if lb_success:
            logging.info(
                "Load balancer is ready. Access it at: http://%s",
current_ip_address
            )
        else:
            logging.error(
                "Couldn't get a successful response from the load balancer
endpoint. Please verify your VPC and security group settings."
            )

    def demo_choices(self) -> None:
        """
        Presents choices for interacting with the deployed service, such as
sending requests to
        the load balancer or checking the health of the targets.
        """
        actions = [
            "Send a GET request to the load balancer endpoint.",
            "Check the health of load balancer targets.",
            "Go to the next part of the demo.",
        ]
        choice = 0
```

```
        while choice != 2:
            logging.info("Choose an action to interact with the service.")
            choice = q.choose("Which action would you like to take? ", actions)
            if choice == 0:
                logging.info("Sending a GET request to the load balancer
endpoint.")
                endpoint =
self.loadbalancer.get_endpoint(self.load_balancer_name)
                logging.info("GET http://%s", endpoint)
                response = requests.get(f"http://{endpoint}")
                logging.info("Response: %s", response.status_code)
                if response.headers.get("content-type") == "application/json":
                    pp(response.json())
            elif choice == 1:
                logging.info("Checking the health of load balancer targets.")
                health =
self.loadbalancer.check_target_health(self.target_group_name)
                for target in health:
                    state = target["TargetHealth"]["State"]
                    logging.info(
                        "Target %s on port %d is %s",
                        target["Target"]["Id"],
                        target["Target"]["Port"],
                        state,
                    )
                    if state != "healthy":
                        logging.warning(
                            "%s: %s",
                            target["TargetHealth"]["Reason"],
                            target["TargetHealth"]["Description"],
                        )
                logging.info(
                    "Note that it can take a minute or two for the health check
to update."
                )
            elif choice == 2:
                logging.info("Proceeding to the next part of the demo.")

    def demo(self) -> None:
        """
        Runs the demonstration, showing how the service responds to different
failure scenarios
        and how a resilient architecture can keep the service running.
        """
```

```
        ssm_only_policy = f"{self.resource_path}/ssm_only_policy.json"

        logging.info("Resetting parameters to starting values for the demo.")
        self.param_helper.reset()

        logging.info(
            "Starting demonstration of the service's resilience under various
failure conditions."
        )
        self.demo_choices()

        logging.info(
            "Simulating failure by changing the Systems Manager parameter to a
non-existent table."
        )
        self.param_helper.put(self.param_helper.table, "this-is-not-a-table")
        logging.info("Sending GET requests will now return failure codes.")
        self.demo_choices()

        logging.info("Switching to static response mode to mitigate failure.")
        self.param_helper.put(self.param_helper.failure_response, "static")
        logging.info("Sending GET requests will now return static responses.")
        self.demo_choices()

        logging.info("Restoring normal operation of the recommendation service.")
        self.param_helper.put(self.param_helper.table,
self.recommendation.table_name)

        logging.info(
            "Introducing a failure by assigning bad credentials to one of the
instances."
        )
        self.autoscaler.create_instance_profile(
            ssm_only_policy,
            self.autoscaler.bad_creds_policy_name,
            self.autoscaler.bad_creds_role_name,
            self.autoscaler.bad_creds_profile_name,
            ["AmazonSSMManagedInstanceCore"],
        )
        instances = self.autoscaler.get_instances()
        bad_instance_id = instances[0]
        instance_profile = self.autoscaler.get_instance_profile(bad_instance_id)
        logging.info(
            "Replacing instance profile with bad credentials for instance %s.",
```

```
            bad_instance_id,
        )
        self.autoscaler.replace_instance_profile(
            bad_instance_id,
            self.autoscaler.bad_creds_profile_name,
            instance_profile["AssociationId"],
        )
        logging.info(
            "Sending GET requests may return either a valid recommendation or a
static response."
        )
        self.demo_choices()

        logging.info("Implementing deep health checks to detect unhealthy
instances.")
        self.param_helper.put(self.param_helper.health_check, "deep")
        logging.info("Checking the health of the load balancer targets.")
        self.demo_choices()

        logging.info(
            "Terminating the unhealthy instance to let the auto scaler replace
it."
        )
        self.autoscaler.terminate_instance(bad_instance_id)
        logging.info("The service remains resilient during instance
replacement.")
        self.demo_choices()

        logging.info("Simulating a complete failure of the recommendation
service.")
        self.param_helper.put(self.param_helper.table, "this-is-not-a-table")
        logging.info(
            "All instances will report as unhealthy, but the service will still
return static responses."
        )
        self.demo_choices()
        self.param_helper.reset()

    def destroy(self, automation=False) -> None:
        """
        Destroys all resources created for the demo, including the load balancer,
Auto Scaling group,
        EC2 instances, and DynamoDB table.
        """
```

```
        logging.info(
            "This concludes the demo. Preparing to clean up all AWS resources
 created during the demo."
        )
        if automation:
            cleanup = True
        else:
            cleanup = q.ask(
                "Do you want to clean up all demo resources? (y/n) ", q.is_yesno
            )

        if cleanup:
            logging.info("Deleting load balancer and related resources.")
            self.loadbalancer.delete_load_balancer(self.load_balancer_name)
            self.loadbalancer.delete_target_group(self.target_group_name)
            self.autoscaler.delete_autoscaling_group(self.autoscaler.group_name)
            self.autoscaler.delete_key_pair()
            self.autoscaler.delete_template()
            self.autoscaler.delete_instance_profile(
                self.autoscaler.bad_creds_profile_name,
                self.autoscaler.bad_creds_role_name,
            )
            logging.info("Deleting DynamoDB table and other resources.")
            self.recommendation.destroy()
        else:
            logging.warning(
                "Resources have not been deleted. Ensure you clean them up
 manually to avoid unexpected charges."
            )


def main() -> None:
    """
    Main function to parse arguments and run the appropriate actions for the
 demo.
    """
    parser = argparse.ArgumentParser()
    parser.add_argument(
        "--action",
        required=True,
        choices=["all", "deploy", "demo", "destroy"],
        help="The action to take for the demo. When 'all' is specified, resources
 are\n"
        "deployed, the demo is run, and resources are destroyed.",
```

```
    )
    parser.add_argument(
        "--resource_path",
        default="../../../scenarios/features/resilient_service/resources",
        help="The path to resource files used by this example, such as IAM
policies and\n"
        "instance scripts.",
    )
    args = parser.parse_args()

    logging.info("Starting the Resilient Service demo.")

    prefix = "doc-example-resilience"

    # Service Clients
    ddb_client = boto3.client("dynamodb")
    elb_client = boto3.client("elbv2")
    autoscaling_client = boto3.client("autoscaling")
    ec2_client = boto3.client("ec2")
    ssm_client = boto3.client("ssm")
    iam_client = boto3.client("iam")

    # Wrapper instantiations
    recommendation = RecommendationService(
        "doc-example-recommendation-service", ddb_client
    )
    autoscaling_wrapper = AutoScalingWrapper(
        prefix,
        "t3.micro",
        "/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2",
        autoscaling_client,
        ec2_client,
        ssm_client,
        iam_client,
    )
    elb_wrapper = ElasticLoadBalancerWrapper(elb_client)
    param_helper = ParameterHelper(recommendation.table_name, ssm_client)

    # Demo invocation
    runner = Runner(
        args.resource_path,
        recommendation,
        autoscaling_wrapper,
        elb_wrapper,
```

```
        param_helper,
    )
    actions = [args.action] if args.action != "all" else ["deploy", "demo",
 "destroy"]
    for action in actions:
        if action == "deploy":
            runner.deploy()
        elif action == "demo":
            runner.demo()
        elif action == "destroy":
            runner.destroy()

    logging.info("Demo completed successfully.")


if __name__ == "__main__":
    logging.basicConfig(level=logging.INFO, format="%(levelname)s: %(message)s")
    main()
```

Cree una clase que agrupe las acciones de escalado automático y Amazon EC2.

```
class AutoScalingWrapper:
    """
    Encapsulates Amazon EC2 Auto Scaling and EC2 management actions.
    """

    def __init__(
        self,
        resource_prefix: str,
        inst_type: str,
        ami_param: str,
        autoscaling_client: boto3.client,
        ec2_client: boto3.client,
        ssm_client: boto3.client,
        iam_client: boto3.client,
    ):
        """
        Initializes the AutoScaler class with the necessary parameters.

        :param resource_prefix: The prefix for naming AWS resources that are
 created by this class.
        :param inst_type: The type of EC2 instance to create, such as t3.micro.
```

```
        :param ami_param: The Systems Manager parameter used to look up the AMI
that is created.
        :param autoscaling_client: A Boto3 EC2 Auto Scaling client.
        :param ec2_client: A Boto3 EC2 client.
        :param ssm_client: A Boto3 Systems Manager client.
        :param iam_client: A Boto3 IAM client.
        """
        self.inst_type = inst_type
        self.ami_param = ami_param
        self.autoscaling_client = autoscaling_client
        self.ec2_client = ec2_client
        self.ssm_client = ssm_client
        self.iam_client = iam_client
        sts_client = boto3.client("sts")
        self.account_id = sts_client.get_caller_identity()["Account"]

        self.key_pair_name = f"{resource_prefix}-key-pair"
        self.launch_template_name = f"{resource_prefix}-template-"
        self.group_name = f"{resource_prefix}-group"

        # Happy path
        self.instance_policy_name = f"{resource_prefix}-pol"
        self.instance_role_name = f"{resource_prefix}-role"
        self.instance_profile_name = f"{resource_prefix}-prof"

        # Failure mode
        self.bad_creds_policy_name = f"{resource_prefix}-bc-pol"
        self.bad_creds_role_name = f"{resource_prefix}-bc-role"
        self.bad_creds_profile_name = f"{resource_prefix}-bc-prof"


    def create_policy(self, policy_file: str, policy_name: str) -> str:
        """
        Creates a new IAM policy or retrieves the ARN of an existing policy.

        :param policy_file: The path to a JSON file that contains the policy
definition.
        :param policy_name: The name to give the created policy.
        :return: The ARN of the created or existing policy.
        """
        with open(policy_file) as file:
            policy_doc = file.read()

        try:
```

```
            response = self.iam_client.create_policy(
                PolicyName=policy_name, PolicyDocument=policy_doc
            )
            policy_arn = response["Policy"]["Arn"]
            log.info(f"Policy '{policy_name}' created successfully. ARN:
 {policy_arn}")
            return policy_arn

        except ClientError as err:
            if err.response["Error"]["Code"] == "EntityAlreadyExists":
                # If the policy already exists, get its ARN
                response = self.iam_client.get_policy(
                    PolicyArn=f"arn:aws:iam::{self.account_id}:policy/
{policy_name}"
                )
                policy_arn = response["Policy"]["Arn"]
                log.info(f"Policy '{policy_name}' already exists. ARN:
 {policy_arn}")
                return policy_arn
            log.error(f"Full error:\n\t{err}")

    def create_role(self, role_name: str, assume_role_doc: dict) -> str:
        """
        Creates a new IAM role or retrieves the ARN of an existing role.

        :param role_name: The name to give the created role.
        :param assume_role_doc: The assume role policy document that specifies
 which
                                entities can assume the role.
        :return: The ARN of the created or existing role.
        """
        try:
            response = self.iam_client.create_role(
                RoleName=role_name,
 AssumeRolePolicyDocument=json.dumps(assume_role_doc)
            )
            role_arn = response["Role"]["Arn"]
            log.info(f"Role '{role_name}' created successfully. ARN: {role_arn}")
            return role_arn

        except ClientError as err:
            if err.response["Error"]["Code"] == "EntityAlreadyExists":
                # If the role already exists, get its ARN
                response = self.iam_client.get_role(RoleName=role_name)
```

```
                role_arn = response["Role"]["Arn"]
                log.info(f"Role '{role_name}' already exists. ARN: {role_arn}")
                return role_arn
            log.error(f"Full error:\n\t{err}")

    def attach_policy(
        self,
        role_name: str,
        policy_arn: str,
        aws_managed_policies: Tuple[str, ...] = (),
    ) -> None:
        """
        Attaches an IAM policy to a role and optionally attaches additional AWS-
managed policies.

        :param role_name: The name of the role to attach the policy to.
        :param policy_arn: The ARN of the policy to attach.
        :param aws_managed_policies: A tuple of AWS-managed policy names to
 attach to the role.
        """
        try:
            self.iam_client.attach_role_policy(RoleName=role_name,
 PolicyArn=policy_arn)
            for aws_policy in aws_managed_policies:
                self.iam_client.attach_role_policy(
                    RoleName=role_name,
                    PolicyArn=f"arn:aws:iam::aws:policy/{aws_policy}",
                )
            log.info(f"Attached policy {policy_arn} to role {role_name}.")
        except ClientError as err:
            log.error(f"Failed to attach policy {policy_arn} to role
 {role_name}.")
            log.error(f"Full error:\n\t{err}")

    def create_instance_profile(
        self,
        policy_file: str,
        policy_name: str,
        role_name: str,
        profile_name: str,
        aws_managed_policies: Tuple[str, ...] = (),
    ) -> str:
        """
```

```
        Creates a policy, role, and profile that is associated with instances
created by
        this class. An instance's associated profile defines a role that is
assumed by the
        instance. The role has attached policies that specify the AWS permissions
granted to
        clients that run on the instance.

        :param policy_file: The name of a JSON file that contains the policy
definition to
                            create and attach to the role.
        :param policy_name: The name to give the created policy.
        :param role_name: The name to give the created role.
        :param profile_name: The name to the created profile.
        :param aws_managed_policies: Additional AWS-managed policies that are
attached to
                                     the role, such as
AmazonSSMManagedInstanceCore to grant
                                     use of Systems Manager to send commands to
the instance.
        :return: The ARN of the profile that is created.
        """
        assume_role_doc = {
            "Version":"2012-10-17",
            "Statement": [
                {
                    "Effect": "Allow",
                    "Principal": {"Service": "ec2.amazonaws.com"},
                    "Action": "sts:AssumeRole",
                }
            ],
        }
        policy_arn = self.create_policy(policy_file, policy_name)
        self.create_role(role_name, assume_role_doc)
        self.attach_policy(role_name, policy_arn, aws_managed_policies)

        try:
            profile_response = self.iam_client.create_instance_profile(
                InstanceProfileName=profile_name
            )
            waiter = self.iam_client.get_waiter("instance_profile_exists")
            waiter.wait(InstanceProfileName=profile_name)
            time.sleep(10)  # wait a little longer
            profile_arn = profile_response["InstanceProfile"]["Arn"]
```

```
            self.iam_client.add_role_to_instance_profile(
                InstanceProfileName=profile_name, RoleName=role_name
            )
            log.info("Created profile %s and added role %s.", profile_name,
role_name)
        except ClientError as err:
            if err.response["Error"]["Code"] == "EntityAlreadyExists":
                prof_response = self.iam_client.get_instance_profile(
                    InstanceProfileName=profile_name
                )
                profile_arn = prof_response["InstanceProfile"]["Arn"]
                log.info(
                    "Instance profile %s already exists, nothing to do.",
profile_name
                )
            log.error(f"Full error:\n\t{err}")
        return profile_arn


    def get_instance_profile(self, instance_id: str) -> Dict[str, Any]:
        """
        Gets data about the profile associated with an instance.

        :param instance_id: The ID of the instance to look up.
        :return: The profile data.
        """
        try:
            response =
self.ec2_client.describe_iam_instance_profile_associations(
                Filters=[{"Name": "instance-id", "Values": [instance_id]}]
            )
            if not response["IamInstanceProfileAssociations"]:
                log.info(f"No instance profile found for instance
{instance_id}.")
            profile_data = response["IamInstanceProfileAssociations"][0]
            log.info(f"Retrieved instance profile for instance {instance_id}.")
            return profile_data
        except ClientError as err:
            log.error(
                f"Failed to retrieve instance profile for instance
{instance_id}."
            )
            error_code = err.response["Error"]["Code"]
            if error_code == "InvalidInstanceID.NotFound":
```

```
                log.error(f"The instance ID '{instance_id}' does not exist.")
            log.error(f"Full error:\n\t{err}")


    def replace_instance_profile(
        self,
        instance_id: str,
        new_instance_profile_name: str,
        profile_association_id: str,
    ) -> None:
        """
        Replaces the profile associated with a running instance. After the
profile is
        replaced, the instance is rebooted to ensure that it uses the new
profile. When
        the instance is ready, Systems Manager is used to restart the Python web
server.

        :param instance_id: The ID of the instance to restart.
        :param new_instance_profile_name: The name of the new profile to
associate with
                                          the specified instance.
        :param profile_association_id: The ID of the existing profile association
for the
                                       instance.
        """
        try:
            self.ec2_client.replace_iam_instance_profile_association(
                IamInstanceProfile={"Name": new_instance_profile_name},
                AssociationId=profile_association_id,
            )
            log.info(
                "Replaced instance profile for association %s with profile %s.",
                profile_association_id,
                new_instance_profile_name,
            )
            time.sleep(5)

            self.ec2_client.reboot_instances(InstanceIds=[instance_id])
            log.info("Rebooting instance %s.", instance_id)
            waiter = self.ec2_client.get_waiter("instance_running")
            log.info("Waiting for instance %s to be running.", instance_id)
            waiter.wait(InstanceIds=[instance_id])
            log.info("Instance %s is now running.", instance_id)
```

```
            self.ssm_client.send_command(
                InstanceIds=[instance_id],
                DocumentName="AWS-RunShellScript",
                Parameters={"commands": ["cd / && sudo python3 server.py 80"]},
            )
            log.info(f"Restarted the Python web server on instance
'{instance_id}'.")
        except ClientError as err:
            log.error("Failed to replace instance profile.")
            error_code = err.response["Error"]["Code"]
            if error_code == "InvalidAssociationID.NotFound":
                log.error(
                    f"Association ID '{profile_association_id}' does not exist."
                    "Please check the association ID and try again."
                )
            if error_code == "InvalidInstanceId":
                log.error(
                    f"The specified instance ID '{instance_id}' does not exist or
is not available for SSM. "
                    f"Please verify the instance ID and try again."
                )
            log.error(f"Full error:\n\t{err}")


    def delete_instance_profile(self, profile_name: str, role_name: str) -> None:
        """
        Detaches a role from an instance profile, detaches policies from the
role,
        and deletes all the resources.

        :param profile_name: The name of the profile to delete.
        :param role_name: The name of the role to delete.
        """
        try:
            self.iam_client.remove_role_from_instance_profile(
                InstanceProfileName=profile_name, RoleName=role_name
            )

self.iam_client.delete_instance_profile(InstanceProfileName=profile_name)
            log.info("Deleted instance profile %s.", profile_name)
            attached_policies = self.iam_client.list_attached_role_policies(
                RoleName=role_name
            )
```

```
            for pol in attached_policies["AttachedPolicies"]:
                self.iam_client.detach_role_policy(
                    RoleName=role_name, PolicyArn=pol["PolicyArn"]
                )
                if not pol["PolicyArn"].startswith("arn:aws:iam::aws"):
                    self.iam_client.delete_policy(PolicyArn=pol["PolicyArn"])
                log.info("Detached and deleted policy %s.", pol["PolicyName"])
            self.iam_client.delete_role(RoleName=role_name)
            log.info("Deleted role %s.", role_name)
        except ClientError as err:
            log.error(
                f"Couldn't delete instance profile {profile_name} or detach "
                f"policies and delete role {role_name}: {err}"
            )
            if err.response["Error"]["Code"] == "NoSuchEntity":
                log.info(
                    "Instance profile %s doesn't exist, nothing to do.",
profile_name
                )


    def create_key_pair(self, key_pair_name: str) -> None:
        """
        Creates a new key pair.

        :param key_pair_name: The name of the key pair to create.
        """
        try:
            response = self.ec2_client.create_key_pair(KeyName=key_pair_name)
            with open(f"{key_pair_name}.pem", "w") as file:
                file.write(response["KeyMaterial"])
            chmod(f"{key_pair_name}.pem", 0o600)
            log.info("Created key pair %s.", key_pair_name)
        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            log.error(f"Failed to create key pair {key_pair_name}.")
            if error_code == "InvalidKeyPair.Duplicate":
                log.error(f"A key pair with the name '{key_pair_name}' already
exists.")
            log.error(f"Full error:\n\t{err}")


    def delete_key_pair(self) -> None:
        """
```

```
        Deletes a key pair.
        """
        try:
            self.ec2_client.delete_key_pair(KeyName=self.key_pair_name)
            remove(f"{self.key_pair_name}.pem")
            log.info("Deleted key pair %s.", self.key_pair_name)
        except ClientError as err:
            log.error(f"Couldn't delete key pair '{self.key_pair_name}'.")
            log.error(f"Full error:\n\t{err}")
        except FileNotFoundError as err:
            log.info("Key pair %s doesn't exist, nothing to do.",
self.key_pair_name)
            log.error(f"Full error:\n\t{err}")


    def create_template(
        self, server_startup_script_file: str, instance_policy_file: str
    ) -> Dict[str, Any]:
        """
        Creates an Amazon EC2 launch template to use with Amazon EC2 Auto
Scaling. The
        launch template specifies a Bash script in its user data field that runs
after
        the instance is started. This script installs Python packages and starts
a
        Python web server on the instance.

        :param server_startup_script_file: The path to a Bash script file that is
run
                                           when an instance starts.
        :param instance_policy_file: The path to a file that defines a
permissions policy
                                     to create and attach to the instance
profile.
        :return: Information about the newly created template.
        """
        template = {}
        try:
            # Create key pair and instance profile
            self.create_key_pair(self.key_pair_name)
            self.create_instance_profile(
                instance_policy_file,
                self.instance_policy_name,
                self.instance_role_name,
```

```
                self.instance_profile_name,
            )

            # Read the startup script
            with open(server_startup_script_file) as file:
                start_server_script = file.read()

            # Get the latest AMI ID
            ami_latest = self.ssm_client.get_parameter(Name=self.ami_param)
            ami_id = ami_latest["Parameter"]["Value"]

            # Create the launch template
            lt_response = self.ec2_client.create_launch_template(
                LaunchTemplateName=self.launch_template_name,
                LaunchTemplateData={
                    "InstanceType": self.inst_type,
                    "ImageId": ami_id,
                    "IamInstanceProfile": {"Name": self.instance_profile_name},
                    "UserData": base64.b64encode(
                        start_server_script.encode(encoding="utf-8")
                    ).decode(encoding="utf-8"),
                    "KeyName": self.key_pair_name,
                },
            )
            template = lt_response["LaunchTemplate"]
            log.info(
                f"Created launch template {self.launch_template_name} for AMI
{ami_id} on {self.inst_type}."
            )
        except ClientError as err:
            log.error(f"Failed to create launch template
{self.launch_template_name}.")
            error_code = err.response["Error"]["Code"]
            if error_code == "InvalidLaunchTemplateName.AlreadyExistsException":
                log.info(
                    f"Launch template {self.launch_template_name} already exists,
nothing to do."
                )
            log.error(f"Full error:\n\t{err}")
        return template


    def delete_template(self):
        """
```

```
        Deletes a launch template.
        """
        try:
            self.ec2_client.delete_launch_template(
                LaunchTemplateName=self.launch_template_name
            )
            self.delete_instance_profile(
                self.instance_profile_name, self.instance_role_name
            )
            log.info("Launch template %s deleted.", self.launch_template_name)
        except ClientError as err:
            if (
                err.response["Error"]["Code"]
                == "InvalidLaunchTemplateName.NotFoundException"
            ):
                log.info(
                    "Launch template %s does not exist, nothing to do.",
                    self.launch_template_name,
                )
            log.error(f"Full error:\n\t{err}")


    def get_availability_zones(self) -> List[str]:
        """
        Gets a list of Availability Zones in the AWS Region of the Amazon EC2
client.

        :return: The list of Availability Zones for the client Region.
        """
        try:
            response = self.ec2_client.describe_availability_zones()
            zones = [zone["ZoneName"] for zone in response["AvailabilityZones"]]
            log.info(f"Retrieved {len(zones)} availability zones: {zones}.")
        except ClientError as err:
            log.error("Failed to retrieve availability zones.")
            log.error(f"Full error:\n\t{err}")
        else:
            return zones


    def create_autoscaling_group(self, group_size: int) -> List[str]:
        """
        Creates an EC2 Auto Scaling group with the specified size.
```

```
        :param group_size: The number of instances to set for the minimum and
maximum in
                           the group.
        :return: The list of Availability Zones specified for the group.
        """
        try:
            zones = self.get_availability_zones()
            self.autoscaling_client.create_auto_scaling_group(
                AutoScalingGroupName=self.group_name,
                AvailabilityZones=zones,
                LaunchTemplate={
                    "LaunchTemplateName": self.launch_template_name,
                    "Version": "$Default",
                },
                MinSize=group_size,
                MaxSize=group_size,
            )
            log.info(
                f"Created EC2 Auto Scaling group {self.group_name} with
availability zones {zones}."
            )
        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            if error_code == "AlreadyExists":
                log.info(
                    f"EC2 Auto Scaling group {self.group_name} already exists,
nothing to do."
                )
            else:
                log.error(f"Failed to create EC2 Auto Scaling group
{self.group_name}.")
                log.error(f"Full error:\n\t{err}")
        else:
            return zones


    def get_instances(self) -> List[str]:
        """
        Gets data about the instances in the EC2 Auto Scaling group.

        :return: A list of instance IDs in the Auto Scaling group.
        """
        try:
            as_response = self.autoscaling_client.describe_auto_scaling_groups(
```

```
                AutoScalingGroupNames=[self.group_name]
            )
            instance_ids = [
                i["InstanceId"]
                for i in as_response["AutoScalingGroups"][0]["Instances"]
            ]
            log.info(
                f"Retrieved {len(instance_ids)} instances for Auto Scaling group
{self.group_name}."
            )
        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            log.error(
                f"Failed to retrieve instances for Auto Scaling group
{self.group_name}."
            )
            if error_code == "ResourceNotFound":
                log.error(f"The Auto Scaling group '{self.group_name}' does not
exist.")
            log.error(f"Full error:\n\t{err}")
        else:
            return instance_ids


    def terminate_instance(self, instance_id: str, decrementsetting=False) ->
None:
        """
        Terminates an instance in an EC2 Auto Scaling group. After an instance is
        terminated, it can no longer be accessed.

        :param instance_id: The ID of the instance to terminate.
        :param decrementsetting: If True, do not replace terminated instances.
        """
        try:
            self.autoscaling_client.terminate_instance_in_auto_scaling_group(
                InstanceId=instance_id,
                ShouldDecrementDesiredCapacity=decrementsetting,
            )
            log.info("Terminated instance %s.", instance_id)

            # Adding a waiter to ensure the instance is terminated
            waiter = self.ec2_client.get_waiter("instance_terminated")
            log.info("Waiting for instance %s to be terminated...", instance_id)
            waiter.wait(InstanceIds=[instance_id])
```

```
            log.info(
                f"Instance '{instance_id}' has been terminated and will be
replaced."
            )

        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            log.error(f"Failed to terminate instance '{instance_id}'.")
            if error_code == "ScalingActivityInProgressFault":
                log.error(
                    "Scaling activity is currently in progress. "
                    "Wait for the scaling activity to complete before attempting
to terminate the instance again."
                )
            elif error_code == "ResourceContentionFault":
                log.error(
                    "The request failed due to a resource contention issue. "
                    "Ensure that no conflicting operations are being performed on
the resource."
                )
            log.error(f"Full error:\n\t{err}")

    def attach_load_balancer_target_group(
        self, lb_target_group: Dict[str, Any]
    ) -> None:
        """
        Attaches an Elastic Load Balancing (ELB) target group to this EC2 Auto
Scaling group.
        The target group specifies how the load balancer forwards requests to the
instances
        in the group.

        :param lb_target_group: Data about the ELB target group to attach.
        """
        try:
            self.autoscaling_client.attach_load_balancer_target_groups(
                AutoScalingGroupName=self.group_name,
                TargetGroupARNs=[lb_target_group["TargetGroupArn"]],
            )
            log.info(
                "Attached load balancer target group %s to auto scaling group
%s.",
                lb_target_group["TargetGroupName"],
                self.group_name,
```

```
            )
        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            log.error(
                f"Failed to attach load balancer target group
'{lb_target_group['TargetGroupName']}'."
            )
            if error_code == "ResourceContentionFault":
                log.error(
                    "The request failed due to a resource contention issue. "
                    "Ensure that no conflicting operations are being performed on
the resource."
                )
            elif error_code == "ServiceLinkedRoleFailure":
                log.error(
                    "The operation failed because the service-linked role is not
ready or does not exist. "
                    "Check that the service-linked role exists and is correctly
configured."
                )
            log.error(f"Full error:\n\t{err}")


    def delete_autoscaling_group(self, group_name: str) -> None:
        """
        Terminates all instances in the group, then deletes the EC2 Auto Scaling
group.

        :param group_name: The name of the group to delete.
        """
        try:
            response = self.autoscaling_client.describe_auto_scaling_groups(
                AutoScalingGroupNames=[group_name]
            )
            groups = response.get("AutoScalingGroups", [])
            if len(groups) > 0:
                self.autoscaling_client.update_auto_scaling_group(
                    AutoScalingGroupName=group_name, MinSize=0
                )
                instance_ids = [inst["InstanceId"] for inst in groups[0]
["Instances"]]
                for inst_id in instance_ids:
                    self.terminate_instance(inst_id)
```

```
                # Wait for all instances to be terminated
                if instance_ids:
                    waiter = self.ec2_client.get_waiter("instance_terminated")
                    log.info("Waiting for all instances to be terminated...")
                    waiter.wait(InstanceIds=instance_ids)
                    log.info("All instances have been terminated.")
            else:
                log.info(f"No groups found named '{group_name}'! Nothing to do.")
        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            log.error(f"Failed to delete Auto Scaling group '{group_name}'.")
            if error_code == "ScalingActivityInProgressFault":
                log.error(
                    "Scaling activity is currently in progress. "
                    "Wait for the scaling activity to complete before attempting
to delete the group again."
                )
            elif error_code == "ResourceContentionFault":
                log.error(
                    "The request failed due to a resource contention issue. "
                    "Ensure that no conflicting operations are being performed on
the group."
                )
            log.error(f"Full error:\n\t{err}")


    def get_default_vpc(self) -> Dict[str, Any]:
        """
        Gets the default VPC for the account.

        :return: Data about the default VPC.
        """
        try:
            response = self.ec2_client.describe_vpcs(
                Filters=[{"Name": "is-default", "Values": ["true"]}]
            )
        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            log.error("Failed to retrieve the default VPC.")
            if error_code == "UnauthorizedOperation":
                log.error(
                    "You do not have the necessary permissions to describe VPCs.
"
```

```
                    "Ensure that your AWS IAM user or role has the correct
 permissions."
                )
            elif error_code == "InvalidParameterValue":
                log.error(
                    "One or more parameters are invalid. Check the request
 parameters."
                )

            log.error(f"Full error:\n\t{err}")
        else:
            if "Vpcs" in response and response["Vpcs"]:
                log.info(f"Retrieved default VPC: {response['Vpcs'][0]
['VpcId']}")
                return response["Vpcs"][0]
            else:
                pass


    def verify_inbound_port(
        self, vpc: Dict[str, Any], port: int, ip_address: str
    ) -> Tuple[Dict[str, Any], bool]:
        """
        Verify the default security group of the specified VPC allows ingress
 from this
        computer. This can be done by allowing ingress from this computer's IP
        address. In some situations, such as connecting from a corporate network,
 you
        must instead specify a prefix list ID. You can also temporarily open the
 port to
        any IP address while running this example. If you do, be sure to remove
 public
        access when you're done.

        :param vpc: The VPC used by this example.
        :param port: The port to verify.
        :param ip_address: This computer's IP address.
        :return: The default security group of the specified VPC, and a value
 that indicates
                 whether the specified port is open.
        """
        try:
            response = self.ec2_client.describe_security_groups(
                Filters=[
```

```
                    {"Name": "group-name", "Values": ["default"]},
                    {"Name": "vpc-id", "Values": [vpc["VpcId"]]},
                ]
            )
            sec_group = response["SecurityGroups"][0]
            port_is_open = False
            log.info(f"Found default security group {sec_group['GroupId']}.")

            for ip_perm in sec_group["IpPermissions"]:
                if ip_perm.get("FromPort", 0) == port:
                    log.info(f"Found inbound rule: {ip_perm}")
                    for ip_range in ip_perm["IpRanges"]:
                        cidr = ip_range.get("CidrIp", "")
                        if cidr.startswith(ip_address) or cidr == "0.0.0.0/0":
                            port_is_open = True
                    if ip_perm["PrefixListIds"]:
                        port_is_open = True
                    if not port_is_open:
                        log.info(
                            f"The inbound rule does not appear to be open to
either this computer's IP "
                            f"address of {ip_address}, to all IP addresses
(0.0.0.0/0), or to a prefix list ID."
                        )
                    else:
                        break
        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            log.error(
                f"Failed to verify inbound rule for port {port} for VPC
{vpc['VpcId']}."
            )
            if error_code == "InvalidVpcID.NotFound":
                log.error(
                    f"The specified VPC ID '{vpc['VpcId']}' does not exist.
Please check the VPC ID."
                )
            log.error(f"Full error:\n\t{err}")
        else:
            return sec_group, port_is_open


    def open_inbound_port(self, sec_group_id: str, port: int, ip_address: str) ->
None:
```

```
        """
        Add an ingress rule to the specified security group that allows access on
the
        specified port from the specified IP address.

        :param sec_group_id: The ID of the security group to modify.
        :param port: The port to open.
        :param ip_address: The IP address that is granted access.
        """
        try:
            self.ec2_client.authorize_security_group_ingress(
                GroupId=sec_group_id,
                CidrIp=f"{ip_address}/32",
                FromPort=port,
                ToPort=port,
                IpProtocol="tcp",
            )
            log.info(
                "Authorized ingress to %s on port %s from %s.",
                sec_group_id,
                port,
                ip_address,
            )
        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            log.error(
                f"Failed to authorize ingress to security group '{sec_group_id}'
on port {port} from {ip_address}."
            )
            if error_code == "InvalidGroupId.Malformed":
                log.error(
                    "The security group ID is malformed. "
                    "Please verify that the security group ID is correct."
                )
            elif error_code == "InvalidPermission.Duplicate":
                log.error(
                    "The specified rule already exists in the security group. "
                    "Check the existing rules for this security group."
                )
            log.error(f"Full error:\n\t{err}")


    def get_subnets(self, vpc_id: str, zones: List[str] = None) -> List[Dict[str,
Any]]:
```

```
        """
        Gets the default subnets in a VPC for a specified list of Availability
Zones.

        :param vpc_id: The ID of the VPC to look up.
        :param zones: The list of Availability Zones to look up.
        :return: The list of subnets found.
        """
        # Ensure that 'zones' is a list, even if None is passed
        if zones is None:
            zones = []
        try:
            paginator = self.ec2_client.get_paginator("describe_subnets")
            page_iterator = paginator.paginate(
                Filters=[
                    {"Name": "vpc-id", "Values": [vpc_id]},
                    {"Name": "availability-zone", "Values": zones},
                    {"Name": "default-for-az", "Values": ["true"]},
                ]
            )

            subnets = []
            for page in page_iterator:
                subnets.extend(page["Subnets"])

            log.info("Found %s subnets for the specified zones.", len(subnets))
            return subnets
        except ClientError as err:
            log.error(
                f"Failed to retrieve subnets for VPC '{vpc_id}' in zones
{zones}."
            )
            error_code = err.response["Error"]["Code"]
            if error_code == "InvalidVpcID.NotFound":
                log.error(
                    "The specified VPC ID does not exist. "
                    "Please check the VPC ID and try again."
                )
            # Add more error-specific handling as needed
            log.error(f"Full error:\n\t{err}")
```

Cree una clase que resuma las acciones de Elastic Load Balancing.

```
class ElasticLoadBalancerWrapper:
    """Encapsulates Elastic Load Balancing (ELB) actions."""

    def __init__(self, elb_client: boto3.client):
        """
        Initializes the LoadBalancer class with the necessary parameters.
        """
        self.elb_client = elb_client


    def create_target_group(
        self, target_group_name: str, protocol: str, port: int, vpc_id: str
    ) -> Dict[str, Any]:
        """
        Creates an Elastic Load Balancing target group. The target group
 specifies how
        the load balancer forwards requests to instances in the group and how
 instance
        health is checked.

        To speed up this demo, the health check is configured with shortened
 times and
        lower thresholds. In production, you might want to decrease the
 sensitivity of
        your health checks to avoid unwanted failures.

        :param target_group_name: The name of the target group to create.
        :param protocol: The protocol to use to forward requests, such as 'HTTP'.
        :param port: The port to use to forward requests, such as 80.
        :param vpc_id: The ID of the VPC in which the load balancer exists.
        :return: Data about the newly created target group.
        """
        try:
            response = self.elb_client.create_target_group(
                Name=target_group_name,
                Protocol=protocol,
                Port=port,
                HealthCheckPath="/healthcheck",
                HealthCheckIntervalSeconds=10,
```

```
                HealthCheckTimeoutSeconds=5,
                HealthyThresholdCount=2,
                UnhealthyThresholdCount=2,
                VpcId=vpc_id,
            )
            target_group = response["TargetGroups"][0]
            log.info(f"Created load balancing target group
'{target_group_name}'.")
            return target_group
        except ClientError as err:
            log.error(
                f"Couldn't create load balancing target group
'{target_group_name}'."
            )
            error_code = err.response["Error"]["Code"]

            if error_code == "DuplicateTargetGroupName":
                log.error(
                    f"Target group name {target_group_name} already exists. "
                    "Check if the target group already exists."
                    "Consider using a different name or deleting the existing
target group if appropriate."
                )
            elif error_code == "TooManyTargetGroups":
                log.error(
                    "Too many target groups exist in the account. "
                    "Consider deleting unused target groups to create space for
new ones."
                )
            log.error(f"Full error:\n\t{err}")


    def delete_target_group(self, target_group_name) -> None:
        """
        Deletes the target group.
        """
        try:
            # Describe the target group to get its ARN
            response =
self.elb_client.describe_target_groups(Names=[target_group_name])
            tg_arn = response["TargetGroups"][0]["TargetGroupArn"]

            # Delete the target group
            self.elb_client.delete_target_group(TargetGroupArn=tg_arn)
```

```
            log.info("Deleted load balancing target group %s.",
target_group_name)

            # Use a custom waiter to wait until the target group is no longer
available
            self.wait_for_target_group_deletion(self.elb_client, tg_arn)
            log.info("Target group %s successfully deleted.", target_group_name)

        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            log.error(f"Failed to delete target group '{target_group_name}'.")
            if error_code == "TargetGroupNotFound":
                log.error(
                    "Load balancer target group either already deleted or never
existed. "
                    "Verify the name and check that the resource exists in the
AWS Console."
                )
            elif error_code == "ResourceInUseException":
                log.error(
                    "Target group still in use by another resource. "
                    "Ensure that the target group is no longer associated with
any load balancers or resources.",
                )
            log.error(f"Full error:\n\t{err}")

    def wait_for_target_group_deletion(
        self, elb_client, target_group_arn, max_attempts=10, delay=30
    ):
        for attempt in range(max_attempts):
            try:

elb_client.describe_target_groups(TargetGroupArns=[target_group_arn])
                print(
                    f"Attempt {attempt + 1}: Target group {target_group_arn}
still exists."
                )
            except ClientError as e:
                if e.response["Error"]["Code"] == "TargetGroupNotFound":
                    print(
                        f"Target group {target_group_arn} has been successfully
deleted."
                    )
                    return
```

```
                else:
                    raise
            time.sleep(delay)
        raise TimeoutError(
            f"Target group {target_group_arn} was not deleted after {max_attempts
* delay} seconds."
        )


   def create_load_balancer(
        self,
        load_balancer_name: str,
        subnet_ids: List[str],
   ) -> Dict[str, Any]:
        """
        Creates an Elastic Load Balancing load balancer that uses the specified
subnets
        and forwards requests to the specified target group.

        :param load_balancer_name: The name of the load balancer to create.
        :param subnet_ids: A list of subnets to associate with the load balancer.
        :return: Data about the newly created load balancer.
        """
        try:
            response = self.elb_client.create_load_balancer(
                Name=load_balancer_name, Subnets=subnet_ids
            )
            load_balancer = response["LoadBalancers"][0]
            log.info(f"Created load balancer '{load_balancer_name}'.")

            waiter = self.elb_client.get_waiter("load_balancer_available")
            log.info(
                f"Waiting for load balancer '{load_balancer_name}' to be
available..."
            )
            waiter.wait(Names=[load_balancer_name])
            log.info(f"Load balancer '{load_balancer_name}' is now available!")

        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            log.error(
                f"Failed to create load balancer '{load_balancer_name}'. Error
code: {error_code}, Message: {err.response['Error']['Message']}"
            )
```

```
            if error_code == "DuplicateLoadBalancerNameException":
                log.error(
                    f"A load balancer with the name '{load_balancer_name}'
already exists. "
                    "Load balancer names must be unique within the AWS region. "
                    "Please choose a different name and try again."
                )
            if error_code == "TooManyLoadBalancersException":
                log.error(
                    "The maximum number of load balancers has been reached in
this account and region. "
                    "You can delete unused load balancers or request an increase
in the service quota from AWS Support."
                )
            log.error(f"Full error:\n\t{err}")
        else:
            return load_balancer


    def create_listener(
        self,
        load_balancer_name: str,
        target_group: Dict[str, Any],
    ) -> Dict[str, Any]:
        """
        Creates a listener for the specified load balancer that forwards requests
to the
        specified target group.

        :param load_balancer_name: The name of the load balancer to create a
listener for.
        :param target_group: An existing target group that is added as a listener
to the
                             load balancer.
        :return: Data about the newly created listener.
        """
        try:
            # Retrieve the load balancer ARN
            load_balancer_response = self.elb_client.describe_load_balancers(
                Names=[load_balancer_name]
            )
            load_balancer_arn = load_balancer_response["LoadBalancers"][0][
                "LoadBalancerArn"
```

```
            ]

            # Create the listener
            response = self.elb_client.create_listener(
                LoadBalancerArn=load_balancer_arn,
                Protocol=target_group["Protocol"],
                Port=target_group["Port"],
                DefaultActions=[
                    {
                        "Type": "forward",
                        "TargetGroupArn": target_group["TargetGroupArn"],
                    }
                ],
            )
            log.info(
                f"Created listener to forward traffic from load balancer
'{load_balancer_name}' to target group '{target_group['TargetGroupName']}'."
            )
            return response["Listeners"][0]
        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            log.error(
                f"Failed to add a listener on '{load_balancer_name}' for target
group '{target_group['TargetGroupName']}'."
            )

            if error_code == "ListenerNotFoundException":
                log.error(
                    f"The listener could not be found for the load balancer
'{load_balancer_name}'. "
                    "Please check the load balancer name and target group
configuration."
                )
            if error_code == "InvalidConfigurationRequestException":
                log.error(
                    f"The configuration provided for the listener on load
balancer '{load_balancer_name}' is invalid. "
                    "Please review the provided protocol, port, and target group
settings."
                )
            log.error(f"Full error:\n\t{err}")


    def delete_load_balancer(self, load_balancer_name) -> None:
```

```
        """
        Deletes a load balancer.

        :param load_balancer_name: The name of the load balancer to delete.
        """
        try:
            response = self.elb_client.describe_load_balancers(
                Names=[load_balancer_name]
            )
            lb_arn = response["LoadBalancers"][0]["LoadBalancerArn"]
            self.elb_client.delete_load_balancer(LoadBalancerArn=lb_arn)
            log.info("Deleted load balancer %s.", load_balancer_name)
            waiter = self.elb_client.get_waiter("load_balancers_deleted")
            log.info("Waiting for load balancer to be deleted...")
            waiter.wait(Names=[load_balancer_name])
        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            log.error(
                f"Couldn't delete load balancer '{load_balancer_name}'. Error
code: {error_code}, Message: {err.response['Error']['Message']}"
            )

            if error_code == "LoadBalancerNotFoundException":
                log.error(
                    f"The load balancer '{load_balancer_name}' does not exist. "
                    "Please check the name and try again."
                )
            log.error(f"Full error:\n\t{err}")


    def get_endpoint(self, load_balancer_name) -> str:
        """
        Gets the HTTP endpoint of the load balancer.

        :return: The endpoint.
        """
        try:
            response = self.elb_client.describe_load_balancers(
                Names=[load_balancer_name]
            )
            return response["LoadBalancers"][0]["DNSName"]
        except ClientError as err:
            log.error(
```

```
                f"Couldn't get the endpoint for load balancer
{load_balancer_name}"
            )
            error_code = err.response["Error"]["Code"]
            if error_code == "LoadBalancerNotFoundException":
                log.error(
                    "Verify load balancer name and ensure it exists in the AWS
console."
                )
            log.error(f"Full error:\n\t{err}")

    @staticmethod
    def verify_load_balancer_endpoint(endpoint) -> bool:
        """
        Verify this computer can successfully send a GET request to the load
balancer endpoint.

        :param endpoint: The endpoint to verify.
        :return: True if the GET request is successful, False otherwise.
        """
        retries = 3
        verified = False
        while not verified and retries > 0:
            try:
                lb_response = requests.get(f"http://{endpoint}")
                log.info(
                    "Got response %s from load balancer endpoint.",
                    lb_response.status_code,
                )
                if lb_response.status_code == 200:
                    verified = True
                else:
                    retries = 0
            except requests.exceptions.ConnectionError:
                log.info(
                    "Got connection error from load balancer endpoint,
retrying..."
                )
                retries -= 1
                time.sleep(10)
        return verified

    def check_target_health(self, target_group_name: str) -> List[Dict[str,
Any]]:
```

```
        """
        Checks the health of the instances in the target group.

        :return: The health status of the target group.
        """
        try:
            tg_response = self.elb_client.describe_target_groups(
                Names=[target_group_name]
            )
            health_response = self.elb_client.describe_target_health(
                TargetGroupArn=tg_response["TargetGroups"][0]["TargetGroupArn"]
            )
        except ClientError as err:
            log.error(f"Couldn't check health of {target_group_name} target(s).")
            error_code = err.response["Error"]["Code"]
            if error_code == "LoadBalancerNotFoundException":
                log.error(
                    "Load balancer associated with the target group was not
found. "
                    "Ensure the load balancer exists, is in the correct AWS
region, and "
                    "that you have the necessary permissions to access it.",
                )
            elif error_code == "TargetGroupNotFoundException":
                log.error(
                    "Target group was not found. "
                    "Verify the target group name, check that it exists in the
correct region, "
                    "and ensure it has not been deleted or created in a different
account.",
                )
            log.error(f"Full error:\n\t{err}")
        else:
            return health_response["TargetHealthDescriptions"]
```

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: str, dynamodb_client: boto3.client):
        """
        Initializes the RecommendationService class with the necessary
parameters.

        :param table_name: The name of the DynamoDB recommendations table.
        :param dynamodb_client: A Boto3 DynamoDB client.
        """
        self.table_name = table_name
        self.dynamodb_client = dynamodb_client

    def create(self) -> Dict[str, Any]:
        """
        Creates a DynamoDB table to use as a recommendation service. The table
has a
        hash key named 'MediaType' that defines the type of media recommended,
such as
        Book or Movie, and a range key named 'ItemId' that, combined with the
MediaType,
        forms a unique identifier for the recommended item.

        :return: Data about the newly created table.
        :raises RecommendationServiceError: If the table creation fails.
        """
        try:
            response = self.dynamodb_client.create_table(
                TableName=self.table_name,
                AttributeDefinitions=[
                    {"AttributeName": "MediaType", "AttributeType": "S"},
                    {"AttributeName": "ItemId", "AttributeType": "N"},
                ],
                KeySchema=[
                    {"AttributeName": "MediaType", "KeyType": "HASH"},
                    {"AttributeName": "ItemId", "KeyType": "RANGE"},
                ],
                ProvisionedThroughput={"ReadCapacityUnits": 5,
"WriteCapacityUnits": 5},
            )
            log.info("Creating table %s...", self.table_name)
```

```
            waiter = self.dynamodb_client.get_waiter("table_exists")
            waiter.wait(TableName=self.table_name)
            log.info("Table %s created.", self.table_name)
        except ClientError as err:
            if err.response["Error"]["Code"] == "ResourceInUseException":
                log.info("Table %s exists, nothing to be done.", self.table_name)
            else:
                raise RecommendationServiceError(
                    self.table_name, f"ClientError when creating table: {err}."
                )
        else:
            return response

    def populate(self, data_file: str) -> None:
        """
        Populates the recommendations table from a JSON file.

        :param data_file: The path to the data file.
        :raises RecommendationServiceError: If the table population fails.
        """
        try:
            with open(data_file) as data:
                items = json.load(data)
            batch = [{"PutRequest": {"Item": item}} for item in items]
            self.dynamodb_client.batch_write_item(RequestItems={self.table_name:
batch})
            log.info(
                "Populated table %s with items from %s.", self.table_name,
data_file
            )
        except ClientError as err:
            raise RecommendationServiceError(
                self.table_name, f"Couldn't populate table from {data_file}:
{err}"
            )

    def destroy(self) -> None:
        """
        Deletes the recommendations table.

        :raises RecommendationServiceError: If the table deletion fails.
        """
        try:
            self.dynamodb_client.delete_table(TableName=self.table_name)
```

```
            log.info("Deleting table %s...", self.table_name)
            waiter = self.dynamodb_client.get_waiter("table_not_exists")
            waiter.wait(TableName=self.table_name)
            log.info("Table %s deleted.", self.table_name)
        except ClientError as err:
            if err.response["Error"]["Code"] == "ResourceNotFoundException":
                log.info("Table %s does not exist, nothing to do.",
 self.table_name)
            else:
                raise RecommendationServiceError(
                    self.table_name, f"ClientError when deleting table: {err}."
                )
```

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: str = "doc-example-resilient-architecture-table"
    failure_response: str = "doc-example-resilient-architecture-failure-response"
    health_check: str = "doc-example-resilient-architecture-health-check"

    def __init__(self, table_name: str, ssm_client: boto3.client):
        """
        Initializes the ParameterHelper class with the necessary parameters.

        :param table_name: The name of the DynamoDB table that is used as a
 recommendation
                           service.
        :param ssm_client: A Boto3 Systems Manager client.
        """
        self.ssm_client = ssm_client
        self.table_name = table_name
```

```
    def reset(self) -> None:
        """
        Resets the Systems Manager parameters to starting values for the demo.
        These are the name of the DynamoDB recommendation table, no response when
a
        dependency fails, and shallow health checks.
        """
        self.put(self.table, self.table_name)
        self.put(self.failure_response, "none")
        self.put(self.health_check, "shallow")

    def put(self, name: str, value: str) -> None:
        """
        Sets the value of a named Systems Manager parameter.

        :param name: The name of the parameter.
        :param value: The new value of the parameter.
        :raises ParameterHelperError: If the parameter value cannot be set.
        """
        try:
            self.ssm_client.put_parameter(
                Name=name, Value=value, Overwrite=True, Type="String"
            )
            log.info("Setting parameter %s to '%s'.", name, value)
        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            log.error(f"Failed to set parameter {name}.")
            if error_code == "ParameterLimitExceeded":
                log.error(
                    "The parameter limit has been exceeded. "
                    "Consider deleting unused parameters or request a limit
increase."
                )
            elif error_code == "ParameterAlreadyExists":
                log.error(
                    "The parameter already exists and overwrite is set to False.
"
                    "Use Overwrite=True to update the parameter."
                )
            log.error(f"Full error:\n\t{err}")
```

- Para 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
  - ReplaceIamInstanceProfileAssociation
  - TerminateInstanceInAutoScalingGroup
  - UpdateAutoScalingGroup

Para obtener una lista completa de guías para desarrolladores del AWS SDK y ejemplos de código, consulteUtilización de este servicio con un AWS SDK. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Solución de problemas de Amazon EC2 Auto Scaling

Amazon EC2 Auto Scaling proporciona errores específicos y descriptivos para ayudarle a solucionar problemas. Puede encontrar los mensajes de error en la descripción de las actividades de escalado.

Temas

- Recuperación de un mensaje de error de las actividades de escalado
- Desactivación de las actividades de escalado
- Recursos adicionales de solución de problemas
- Solución de problemas de Amazon EC2 Auto Scaling: errores de lanzamiento de instancias de EC2
- Solución de problemas de Amazon EC2 Auto Scaling: problemas de AMI
- Solución de problemas de Amazon EC2 Auto Scaling: problemas del equilibrador de carga
- Solución de problemas de Amazon EC2 Auto Scaling: plantillas de lanzamiento

Recuperación de un mensaje de error de las actividades de escalado

Para recuperar un mensaje de error de la descripción de las actividades de escalado, utilice el comando describe-scaling-activities. Tiene un registro de actividades de escalado que se remonta a 6 semanas atrás. Las actividades de escalado se ordenan por hora de inicio, enumerando primero las actividades de escalado más recientes.

> **Note**
>
> Las actividades de escalado también se muestran en el historial de actividad de la consola de Amazon EC2 Auto Scaling en la pestaña Activity (Actividad) del grupo de Auto Scaling.

Para ver las actividades de escalado de un grupo de Auto Scaling específico, utilice el siguiente comando.

```
aws autoscaling describe-scaling-activities --auto-scaling-group-name my-asg
```

A continuación, se muestra un ejemplo de respuesta, donde `StatusCode` contiene el estado actual de la actividad y `StatusMessage` contiene el mensaje de error.

```
{
    "Activities": [
        {
            "ActivityId": "3b05dbf6-037c-b92f-133f-38275269dc0f",
            "AutoScalingGroupName": "my-asg",
            "Description": "Launching a new EC2 instance: i-003a5b3ffe1e9358e.  Status
 Reason: Instance failed to complete user's Lifecycle Action: Lifecycle Action with
 token e85eb647-4fe0-4909-b341-a6c42d8aba1f was abandoned: Lifecycle Action Completed
 with ABANDON Result",
            "Cause": "At 2021-01-11T00:35:52Z a user request created an
 AutoScalingGroup changing the desired capacity from 0 to 1.  At 2021-01-11T00:35:53Z
 an instance was started in response to a difference between desired and actual
 capacity, increasing the capacity from 0 to 1.",
            "StartTime": "2021-01-11T00:35:55.542Z",
            "EndTime": "2021-01-11T01:06:31Z",
            "StatusCode": "Cancelled",
            "StatusMessage": "Instance failed to complete user's Lifecycle Action:
 Lifecycle Action with token e85eb647-4fe0-4909-b341-a6c42d8aba1f was abandoned:
 Lifecycle Action Completed with ABANDON Result",
            "Progress": 100,
            "Details": "{\"Subnet ID\":\"subnet-5ea0c127\",\"Availability Zone\":\"us-
west-2b\"...}",
            "AutoScalingGroupARN": "arn:aws:autoscaling:us-
west-2:123456789012:autoScalingGroup:283179a2-
f3ce-423d-93f6-66bb518232f7:autoScalingGroupName/my-asg"
        },
     ...
    ]
}
```

Para obtener una descripción de los campos de la salida, consulte Actividad en la Referencia de API de Amazon EC2 Auto Scaling.

Para ver las actividades de escalado de un grupo eliminado

Para ver las actividades de escalado después de eliminar el grupo Auto Scaling, añada la `--include-deleted-groups` opción al describe-scaling-activitiescomando de la siguiente manera.

```
aws autoscaling describe-scaling-activities --auto-scaling-group-name my-asg --include-
deleted-groups
```

A continuación, se muestra un ejemplo de respuesta, con una actividad de escalado para un grupo eliminado.

```
{
    "Activities": [
        {
            "ActivityId": "e1f5de0e-f93e-1417-34ac-092a76fba220",
            "AutoScalingGroupName": "my-asg",
            "Description": "Launching a new EC2 instance.  Status Reason: Your Spot
 request price of 0.001 is lower than the minimum required Spot request fulfillment
 price of 0.0031. Launching EC2 instance failed.",
            "Cause": "At 2021-01-13T20:47:24Z a user request update of AutoScalingGroup
 constraints to min: 1, max: 5, desired: 3 changing the desired capacity from 0 to 3.
  At 2021-01-13T20:47:27Z an instance was started in response to a difference between
 desired and actual capacity, increasing the capacity from 0 to 3.",
            "StartTime": "2021-01-13T20:47:30.094Z",
            "EndTime": "2021-01-13T20:47:30Z",
            "StatusCode": "Failed",
            "StatusMessage": "Your Spot request price of 0.001 is lower than the
 minimum required Spot request fulfillment price of 0.0031. Launching EC2 instance
 failed.",
            "Progress": 100,
            "Details": "{\"Subnet ID\":\"subnet-5ea0c127\",\"Availability Zone\":\"us-
west-2b\"...}",
            "AutoScalingGroupState": "Deleted",
            "AutoScalingGroupARN": "arn:aws:autoscaling:us-
west-2:123456789012:autoScalingGroup:283179a2-
f3ce-423d-93f6-66bb518232f7:autoScalingGroupName/my-asg"
        },
     ...
    ]
}
```

Desactivación de las actividades de escalado

Dispone de las siguientes opciones si necesita investigar un problema sin que las políticas de escalado o las acciones programadas interfieran:

- Impida que todas las políticas de escalado dinámico y las acciones programadas modifiquen la capacidad deseada del grupo al suspender los procesos `AlarmNotification` y `ScheduledActions`. Para obtener más información, consulte Suspensión y reanudación de procesos de Amazon EC2 Auto Scaling.
- Deshabilite las políticas de escalado dinámico individuales para que no cambien la capacidad deseada del grupo en respuesta a los cambios en la carga. Para obtener más información, consulte Desactivación de una política de escalado para un grupo de escalado automático.
- Actualice las políticas de escalado de seguimiento de destino individuales para que solo se escale horizontalmente (agregue capacidad) al deshabilitar la parte de reducción horizontal de la política. Este método evita que se reduzca la capacidad deseada del grupo, pero permite aumentarla cuando aumenta la carga. Para obtener más información, consulte Políticas de escalado de seguimiento de destino para Amazon EC2 Auto Scaling.
- Actualice su política de escalado predictivo al modo de solo previsión. Mientras esté en el modo de solo previsión, el escalado predictivo seguirá generando previsiones, pero no aumentará automáticamente la capacidad. Para obtener más información, consulte Creación de una política de escalado predictivo para un grupo de escalado automático.

Recursos adicionales de solución de problemas

En las páginas siguientes se proporciona información adicional para solucionar problemas de Amazon EC2 Auto Scaling.

- Verificación de una actividad de escalado para un grupo de escalado automático
- Visualización de gráficos de supervisión en la consola de Amazon EC2 Auto Scaling
- Comprobaciones de estado para instancias en un grupo de escalado automático
- Consideraciones y limitaciones de enlaces de ciclo de vida
- Cómo completar una acción del ciclo de vida en un grupo de escalado automático
- Proporcionar conectividad de red para sus instancias de Auto Scaling mediante Amazon VPC
- Eliminación temporal de las instancias de un grupo de escalado automático
- Desactivación de una política de escalado para un grupo de escalado automático
- Suspensión y reanudación de procesos de Amazon EC2 Auto Scaling
- Control de las instancias de Auto Scaling que se terminan durante una reducción horizontal
- Eliminación de la infraestructura de Auto Scaling

- Cuotas para los recursos y grupos de escalado automático

Los siguientes AWS recursos también pueden ser útiles:

- Temas de Amazon EC2 Auto Scaling en AWS el Knowledge Center
- Preguntas sobre AWS Amazon EC2 Auto Scaling sobre Re:post
- Publicaciones de Amazon EC2 Auto Scaling en AWS el blog de informática
- Solución de problemas CloudFormation en la guía del CloudFormation usuario

A menudo, la solución de problemas requiere consultas y descubrimiento iterativos por parte de un experto o de una comunidad de ayudantes. Si sigue teniendo problemas después de probar las sugerencias de esta sección, póngase en contacto con AWS Support (en Support Consola de administración de AWS, Support Center) o haga una pregunta en AWS Re:post utilizando la etiqueta Amazon EC2 Auto Scaling.

Solución de problemas de Amazon EC2 Auto Scaling: errores de lanzamiento de instancias de EC2

En esta página se proporciona información acerca de las instancias EC2 que no se pueden lanzar, las causas posibles y los pasos que puede realizar para resolver el problema.

Para recuperar un mensaje de error, consulte Recuperación de un mensaje de error de las actividades de escalado.

Cuando su instancia EC2 no se puede lanzar, es posible que aparezca un mensaje de error similar a los siguientes:

Problemas de lanzamiento

- La configuración solicitada no se admite actualmente.
- El grupo de seguridad <nombre del grupo de seguridad > no existe. El lanzamiento de la instancia EC2 ha producido un error.
- El par de claves <par de claves asociado a la instancia EC2> no existe. El lanzamiento de la instancia EC2 ha producido un error.
- el tipo de instancia solicitado (<tipo de instancia>) ya no es compatible con la zona de disponibilidad solicitada (<zona de disponibilidad de la instancia>)...

- Su precio de solicitud de spot de 0,015 es menor que el precio mínimo requerido de cumplimiento de solicitud de spot de 0,0735...
- Nombre de dispositivo inválido <nombre de dispositivo>/Carga de nombre de dispositivo inválido. El lanzamiento de la instancia EC2 ha producido un error.
- El valor (<nombre asociado al dispositivo de almacenamiento de la instancia>) del parámetro virtualName no es válido... El lanzamiento de la instancia EC2 ha producido un error.
- Los mapeos de dispositivos de bloques de EBS no son compatibles con instance-store. AMIs
- Los grupos de ubicación no se pueden utilizar con instancias de tipo “<tipo de instancia>”. El lanzamiento de la instancia EC2 ha producido un error.
- Cliente. InternalError: Error del cliente al iniciarse.
- En la actualidad, no dispone de suficiente capacidad de <tipo de instancia> en la zona de disponibilidad que ha solicitado... El lanzamiento de la instancia EC2 ha producido un error.
- La reserva solicitada no tiene suficiente capacidad compatible y disponible para esta solicitud. El lanzamiento de la instancia EC2 ha producido un error.
- Su reserva de bloques de capacidad <id de reserva>aún no está activa. El lanzamiento de la instancia EC2 ha producido un error.
- No hay capacidad de spot disponible que coincida con su solicitud. El lanzamiento de la instancia EC2 ha producido un error.
- Ya se están ejecutando <número de instancias> instancias. El lanzamiento de la instancia EC2 ha producido un error.

La configuración solicitada no se admite actualmente.

Causa: es posible que algunas opciones de la plantilla de lanzamiento o la configuración de lanzamiento no sean compatibles con el tipo de instancia, o que la configuración de la instancia no sea compatible con la AWS región o las zonas de disponibilidad solicitadas.

Solución: pruebe una configuración de instancia diferente. Para buscar un tipo de instancias que cumpla sus requisitos, consulte Finding an Amazon EC2 instance type en la Guía del usuario de Amazon EC2.

Para obtener más instrucciones para resolver este problema, verifique lo siguiente:

- Asegúrese de haber elegido una AMI compatible con su tipo de instancia. Por ejemplo, si el tipo de instancia utiliza un procesador AWS Graviton basado en ARM en lugar de un procesador Intel

Xeon, necesitará una AMI compatible con ARM. Para obtener más información sobre cómo se elige un tipo de instancia compatible, consulte Compatibilidad para cambiar el tipo de instancia en la Guía del usuario de Amazon EC2.

- Compruebe que el tipo de instancias está disponible en las zonas de disponibilidad y región solicitadas. Es posible que los tipos de instancias de última generación aún no estén disponibles en una región o zona de disponibilidad determinada. Es posible que los tipos de instancias de generación anterior no estén disponibles en las regiones y zonas de disponibilidad más recientes. Para buscar tipos de instancia ofrecidos por ubicación (Región o Zona de disponibilidad), use el comando describe-instance-type-offerings. Para obtener más información, consulte Finding an Amazon EC2 instance type en la Guía del usuario de Amazon EC2.
- Si utiliza instancias dedicadas o hosts dedicados, asegúrese de haber elegido un tipo de instancias compatible con una instancia dedicada o host dedicado.

El grupo de seguridad <nombre del grupo de seguridad > no existe. El lanzamiento de la instancia EC2 ha producido un error.

Causa: Puede que se haya eliminado el grupo de seguridad especificado en la plantilla o configuración de lanzamiento.

Solución:

1. Utilice el comando describe-security-groups para obtener la lista de los grupos de seguridad asociados a su cuenta.
2. En la lista, seleccione los grupos de seguridad que desea usar. Si lo que desea es crear un grupo de seguridad, utilice el comando create-security-group.
3. Cree una nueva configuración de lanzamiento o a plantilla de lanzamiento.
4. Actualice el grupo de Auto Scaling con la nueva plantilla de lanzamiento o configuración de lanzamiento usando el comando update-auto-scaling-group.

El par de claves <par de claves asociado a la instancia EC2> no existe. El lanzamiento de la instancia EC2 ha producido un error.

Causa: puede que se haya eliminado el par de claves al lanzar la instancia.

Solución:

1. Utilice el comando describe-key-pairs para obtener la lista de los pares de claves disponibles.
2. En la lista, seleccione el par de claves que desea usar. Si lo que desea es crear un par de claves, utilice el comando create-key-pair.
3. Cree una nueva configuración de lanzamiento o plantilla de lanzamiento.
4. Actualice el grupo de Auto Scaling con la nueva plantilla de lanzamiento o configuración de lanzamiento usando el comando update-auto-scaling-group.

el tipo de instancia solicitado (<tipo de instancia>) ya no es compatible con la zona de disponibilidad solicitada (<zona de disponibilidad de la instancia>)...

Mensaje de error: el tipo de instancia solicitado (<tipo de instancia>) no es compatible con la zona de disponibilidad solicitada (<zona de disponibilidad de la instancia>). El lanzamiento de la instancia EC2 ha producido un error.

Causa: las zonas de disponibilidad especificadas en el grupo de escalado automático no son compatibles con el tipo de instancia elegido.

Solución:

1. Compruebe qué zonas de disponibilidad son compatibles con el tipo de instancia elegido mediante el describe-instance-type-offeringscomando o desde la consola Amazon EC2 comprobando el valor de las zonas de disponibilidad en el panel de redes de la página de tipos de instancias.
2. Actualice o elimine la subred de cualquier zona no compatible en la configuración de su grupo de Auto Scaling mediante el update-auto-scaling-groupcomando. Para obtener más información, consulte Agregar una zona de disponibilidad.

Su precio de solicitud de spot de 0,015 es menor que el precio mínimo requerido de cumplimiento de solicitud de spot de 0,0735...

Causa: El precio máximo de spot de la solicitud es inferior al precio de spot del tipo de instancia seleccionado.

Solución: Envíe una nueva solicitud con un precio máximo de spot (posiblemente el precio en diferido). Anteriormente, el precio de spot que se pagaba se basaba en pujas. Hoy, se paga el precio

de spot actual. Al establecer el precio máximo más alto, ofrece al servicio de spot de Amazon EC2 una mejor oportunidad de lanzar y mantener la cantidad de capacidad requerida.

Nombre de dispositivo inválido <nombre de dispositivo>/Carga de nombre de dispositivo inválido. El lanzamiento de la instancia EC2 ha producido un error.

Causa 1: las asignaciones de dispositivos de bloques de la plantilla de lanzamiento o configuración de lanzamiento podrían contener nombres de dispositivos de bloques que no estén disponibles o que no se admitan actualmente.

Solución:

1. Compruebe qué nombres de dispositivos están disponibles para la configuración de su instancia específica. Para obtener más detalles sobre la asignación de nombres de dispositivos, consulte Device names on Linux instances en la Guía del usuario de Amazon EC2.
2. Cree manualmente una instancia de Amazon EC2 que no forme parte del grupo de Auto Scaling e investigue el problema. Si la configuración de asignación de nombres de los dispositivos de bloque entra en conflicto con los de la imagen de máquina de Amazon (AMI), se producirá un error en la instancia durante el lanzamiento. Para obtener más información, consulte Mapeos de dispositivos de bloques en la Guía del usuario de Amazon EC2.
3. Después de confirmar que su instancia se lanzó correctamente, utilice el comando describe-volumes y vea cómo se exponen los volúmenes a la instancia.
4. Cree una nueva plantilla de lanzamiento o configuración de lanzamiento utilizando el nombre del dispositivo que se muestra en la descripción del volumen.
5. Actualice el grupo de Auto Scaling con la nueva plantilla de lanzamiento o configuración de lanzamiento usando el comando update-auto-scaling-group.

El valor (<nombre asociado al dispositivo de almacenamiento de la instancia>) del parámetro virtualName no es válido... El lanzamiento de la instancia EC2 ha producido un error.

Causa: el formato especificado para el nombre virtual asociado al dispositivo de bloques es incorrecto.

Solución:

1. Cree una nueva plantilla de lanzamiento o configuración de lanzamiento especificando el nombre del dispositivo en el parámetro `virtualName`. Para obtener información sobre el formato de los nombres de dispositivos, consulte Device naming on Linux instances en la Guía del usuario de Amazon EC2.
2. Actualice el grupo de Auto Scaling con la nueva plantilla de lanzamiento o configuración de lanzamiento usando el comando update-auto-scaling-group.

Los mapeos de dispositivos de bloques de EBS no son compatibles con instance-store. AMIs

Causa: Las asignaciones de dispositivos de bloques especificadas en la plantilla de lanzamiento o configuración de lanzamiento no se admiten en la instancia.

Solución:

1. Cree una nueva plantilla de lanzamiento o configuración de lanzamiento con asignaciones de dispositivos de bloques compatibles con el tipo de instancia. Para obtener más información, consulte Asignación de dispositivos de bloques en la Guía del usuario de Amazon EC2.
2. Actualice el grupo de Auto Scaling con la nueva plantilla de lanzamiento o configuración de lanzamiento usando el comando update-auto-scaling-group.

Los grupos de ubicación no se pueden utilizar con instancias de tipo “<tipo de instancia>”. El lanzamiento de la instancia EC2 ha producido un error.

Causa: el grupo de ubicación en clúster contiene un tipo de instancia no válido.

Solución:

1. Para obtener información sobre los tipos de instancias válidos admitidos por los grupos de ubicación, consulte Grupos de ubicación en la Guía del usuario de Amazon EC2.
2. Siga las instrucciones que se detallan en Grupos de ubicación para crear un nuevo grupo de ubicación.
3. Otra opción es crear una nueva plantilla de lanzamiento o configuración de lanzamiento con el tipo de instancia compatible.
4. Actualice el grupo de Auto Scaling con un nuevo grupo de ubicación, plantilla de lanzamiento o configuración de lanzamiento utilizando el comando update-auto-scaling-group.

Cliente. InternalError: Error del cliente al iniciarse.

Problema: Amazon EC2 Auto Scaling intenta lanzar una instancia que tiene un volumen de EBS cifrado, pero el rol vinculado al servicio no tiene acceso a la clave administrada por AWS KMS el cliente que se utiliza para cifrarlo. Para obtener más información, consulte Política de AWS KMS claves obligatoria para su uso con volúmenes cifrados.

Causa 1: Necesita una política clave que dé permiso para usar la clave administrada por el cliente al rol vinculado al servicio adecuado.

Solución 1: Permita que el rol vinculado al servicio use la clave administrada por el cliente tal y como se indica a continuación:

1. Determine qué función vinculada al servicio se va a utilizar para este grupo de Auto Scaling.
2. Actualice la política de claves en la clave administrada por el cliente y permita que el rol vinculado al servicio use la clave administrada por el cliente.
3. Actualice el grupo de Auto Scaling para que utilice el rol vinculado al servicio.

Para obtener un ejemplo de política de claves que permite que el rol vinculado al servicio utilice la clave administrada por el cliente, consulte Ejemplo 1: secciones de la política de claves que permiten el acceso a la clave administrada por el cliente.

Causa 2: Si la clave administrada por el cliente y el grupo de Auto Scaling están en AWS cuentas diferentes, debe configurar el acceso entre cuentas a la clave administrada por el cliente para dar permiso para usar la clave administrada por el cliente para el rol correspondiente vinculado al servicio.

Solución 2: Permita que el rol vinculado al servicio en la cuenta externa utilice el rol administrado por el cliente en la cuenta local de la siguiente manera:

1. Actualice la política de claves en la clave administrada por el cliente para permitir que la cuenta del grupo de Auto Scaling acceda a la clave administrada por el cliente.
2. Defina un usuario o rol de IAM en la cuenta del grupo de Auto Scaling que pueda conceder permisos.
3. Determine qué función vinculada al servicio se va a utilizar para este grupo de Auto Scaling.
4. Conceda permisos a la clave administrada por el cliente con el rol vinculado al servicio adecuado como la entidad principal beneficiaria.

5. Actualice el grupo de Auto Scaling para que utilice el rol vinculado al servicio.

Para obtener más información, consulte Ejemplo 2: secciones de la política de claves que permiten el acceso entre cuentas a la clave administrada por el cliente.

Solución 3: Utilice una clave administrada por el cliente en la misma cuenta de AWS que el grupo de Auto Scaling.

1. Copie y vuelva a cifrar la instantánea con otra clave administrada por el cliente que pertenezca a la misma cuenta que el grupo de Auto Scaling.
2. Permita que el rol vinculado al servicio use la nueva clave administrada por el cliente. Consulte los pasos de la Solución 1.

En la actualidad, no dispone de suficiente capacidad de <tipo de instancia> en la zona de disponibilidad que ha solicitado... El lanzamiento de la instancia EC2 ha producido un error.

Mensaje de error: En la actualidad, no se dispone de suficiente capacidad de <tipo de instancia> en la zona de disponibilidad solicitada (<zona de disponibilidad solicitada>). Estamos trabajando para aprovisionar capacidad adicional. Puede obtener capacidad para <tipo de instancia> sin especificar una zona de disponibilidad en la solicitud o eligiendo <lista de zonas de disponibilidad que admite actualmente el tipo de instancia>. El lanzamiento de la instancia EC2 ha producido un error.

Causa: en este momento, no se admite la combinación de tipo de instancia y zona de disponibilidad solicitada.

Solucióon: para resolver este problema, pruebe lo siguiente:

- Espere unos minutos a que Amazon EC2 Auto Scaling encuentre capacidad para este tipo de instancia en otras zonas de disponibilidad habilitadas.
- Expanda su grupo de escalado automático a zonas de disponibilidad adicionales. Para obtener más información, consulte Agregar una zona de disponibilidad.
- Siga la práctica recomendada de utilizar un conjunto diverso de tipos de instancia para que no dependa de un tipo de instancia específico. Para obtener más información, consulte Grupos de Auto Scaling con varios tipos de instancia y opciones de compra.

La reserva solicitada no tiene suficiente capacidad compatible y disponible para esta solicitud. El lanzamiento de la instancia EC2 ha producido un error.

Causa 1: ha alcanzado el límite del número de instancias que puede lanzar con una reserva de capacidad `targeted` bajo demanda.

Solución 1: aumente la cantidad de instancias que puede lanzar con la reserva de capacidad `targeted` bajo demanda o utilice un grupo de reservas de capacidad para que cualquier instancia que supere la capacidad reservada se lance como capacidad bajo demanda normal. Para obtener más información, consulte Capacidad reservada en zonas de disponibilidad específicas con reserva de capacidad.

Causa 2: ha alcanzado el límite del número de instancias que puede lanzar en un bloque de capacidad.

Con los bloques de capacidad, está limitado por la cantidad de capacidad que haya adquirido originalmente. Si experimenta un número de lanzamientos superior al previsto y utiliza toda la capacidad que tiene disponible, esto provoca que los lanzamientos fallen. Las instancias de finalización pasan por un largo proceso de limpieza antes de que finalicen por completo. Durante este tiempo, no se pueden volver a usar. Esto también puede provocar errores de lanzamiento. Para obtener más información, consulte Uso Bloques de capacidad para cargas de trabajo de aprendizaje automático.

Solución 2 para resolver este problema, pruebe lo siguiente:

- Mantenga la solicitud tal cual. Si una instancia de bloque de capacidad está finalizando, debe esperar varios minutos a que la instancia acabe de finalizar y la capacidad vuelva a estar disponible. Amazon EC2 Auto Scaling sigue realizando la solicitud de lanzamiento automáticamente hasta que haya capacidad disponible.
- Asegúrese de adquirir capacidad suficiente para adaptarse a los picos de carga de trabajo, de modo que no se produzca este error con frecuencia.

Su reserva de bloques de capacidad <id de reserva>aún no está activa. El lanzamiento de la instancia EC2 ha producido un error.

Causa: el bloque de capacidad especificado aún no está activo.

Solución: siga el enfoque recomendado para los bloques de capacidad y utilice el escalado programado. Hacerlo le ayuda a asegurarse de aumentar la capacidad deseada de su grupo de escalado automático solo cuando la reserva esté activa y a disminuirla antes de que finalice la reserva.

No hay capacidad de spot disponible que coincida con su solicitud. El lanzamiento de la instancia EC2 ha producido un error.

Causa: En este momento, no hay suficiente capacidad adicional para satisfacer su solicitud de instancias de spot.

Solución: para resolver este problema, pruebe lo siguiente:

- Espere unos minutos; la capacidad puede cambiar frecuentemente. Amazon EC2 Auto Scaling sigue realizando la solicitud de lanzamiento automáticamente hasta que haya capacidad disponible.
- Expanda su grupo de escalado automático a zonas de disponibilidad adicionales. Para obtener más información, consulte Agregar una zona de disponibilidad.
- Siga la práctica recomendada de utilizar un conjunto diverso de tipos de instancia para que no dependa de un tipo de instancia específico. Para obtener más información, consulte Grupos de Auto Scaling con varios tipos de instancia y opciones de compra.

Ya se están ejecutando <número de instancias> instancias. El lanzamiento de la instancia EC2 ha producido un error.

Causa: Ha alcanzado el límite del número de instancias que puede lanzar en una región. Al crear su AWS cuenta, establecemos límites predeterminados en cuanto al número de instancias que puede ejecutar por región.

Solución: para resolver este problema, pruebe lo siguiente:

- Si los límites actuales no son adecuados para sus necesidades, puede solicitar un aumento de cuota por región. Para obtener más información, consulte Cuotas de servicio de Amazon EC2 en la Guía del usuario de Amazon EC2.
- Envíe una nueva solicitud con un número de instancias reducido (que puede aumentar en una fase posterior).

Solución de problemas de Amazon EC2 Auto Scaling: problemas de AMI

Esta página proporciona información sobre los problemas asociados a sus problemas AMIs, sus posibles causas y las medidas que puede tomar para resolverlos.

Para recuperar un mensaje de error, consulte Recuperación de un mensaje de error de las actividades de escalado.

Cuando las instancias EC2 no se pueden lanzar debido a problemas con su AMI, puede recibir uno o varios mensajes de error similares a los siguientes.

Problemas con las AMI

- El ID de AMI <ID de la AMI> no existe. El lanzamiento de la instancia EC2 ha producido un error.
- La AMI <ID de AMI> está pendiente y no se puede ejecutar. El lanzamiento de la instancia EC2 ha producido un error.
- Nombre de dispositivo no válido <nombre de dispositivo>. El lanzamiento de la instancia EC2 ha producido un error.
- La arquitectura “arm64” del tipo de instancia especificado no coincide con la arquitectura “x86_64” de la AMI especificada... Falló el lanzamiento de la instancia de EC2.
- La AMI “<ID de AMI>” está deshabilitada y no se puede ejecutar. El lanzamiento de la instancia EC2 ha producido un error.

> **Important**
>
> AWS permite compartir una AMI de forma privada con otra AWS cuenta mediante la modificación de los permisos de la AMI. Si una AMI se convierte en privada sin compartirla, se puede producir un error de autorización al lanzar nuevas instancias. Para obtener más información sobre cómo compartir información privada AMIs, consulte Compartir una AMI con AWS cuentas específicas en la Guía del usuario de Amazon EC2.

El ID de AMI <ID de la AMI> no existe. El lanzamiento de la instancia EC2 ha producido un error.

- Causa: Puede que se haya eliminado la AMI después de crear la plantilla de lanzamiento o configuración de lanzamiento.
- Solución:
  1. Cree una nueva plantilla de lanzamiento o configuración de lanzamiento con una AMI válida.
  2. Actualice el grupo de Auto Scaling con la nueva plantilla de lanzamiento o configuración de lanzamiento usando el comando update-auto-scaling-group.

La AMI <ID de AMI> está pendiente y no se puede ejecutar. El lanzamiento de la instancia EC2 ha producido un error.

Causa: es posible que la AMI acabe de crearse (a partir de una instantánea de una instancia en ejecución o de otra manera) y que aún no esté disponible.

Solución: Debe esperar a que la AMI esté disponible y, a continuación, crear la plantilla de lanzamiento o configuración de lanzamiento.

Nombre de dispositivo no válido <nombre de dispositivo>. El lanzamiento de la instancia EC2 ha producido un error.

Causa: al adjuntar un volumen de EBS a una instancia EC2, debe proporcionar un nombre de dispositivo válido para el volumen. La AMI seleccionada debe admitir este nombre de dispositivo.

Solución:

1. Cree una nueva plantilla de lanzamiento o configuración de lanzamiento y especifique el nombre correcto del dispositivo para su AMI. La convención de nomenclatura recomendada varía según el tipo de virtualización de la AMI. Para obtener más información, consulte Nombres de dispositivos en la Guía del usuario de Amazon EC2.
2. Actualice el grupo de Auto Scaling con la nueva plantilla de lanzamiento o configuración de lanzamiento usando el comando update-auto-scaling-group.

La arquitectura “arm64” del tipo de instancia especificado no coincide con la arquitectura “x86_64” de la AMI especificada... Falló el lanzamiento de la instancia de EC2.

Causa 1: si la arquitectura de la AMI y el tipo de instancia utilizado en la plantilla de lanzamiento o la configuración de lanzamiento no son los mismos, se produce un error cuando Amazon EC2 Auto Scaling intenta lanzar una instancia con una configuración de instancias incompatible.

Solución 1:

1. Compruebe la arquitectura de la AMI mediante el comando describe-images o desde la consola Amazon EC2 comprobando el valor de la arquitectura en el panel de detalles de la página Amazon Machine Images (). AMIs
2. Busque un tipo de instancia que tenga la misma arquitectura que su AMI mediante el describe-instance-typescomando o desde la consola Amazon EC2 consultando la columna Arquitectura de la pantalla Tipos de instancias. Para obtener más información sobre cómo se elige un tipo de instancia compatible, consulte Compatibilidad para cambiar el tipo de instancia en la Guía del usuario de Amazon EC2.
3. Cree una nueva plantilla de lanzamiento o configuración de lanzamiento con un tipo de instancia que tenga la misma arquitectura que la AMI.
4. Actualice el grupo de Auto Scaling con la nueva plantilla de lanzamiento o configuración de lanzamiento usando el comando update-auto-scaling-group.

Causa 2: Amazon EC2 Auto Scaling intenta lanzar un tipo de instancia especificado en la política de instancias mixtas del grupo de escalado automático, pero el tipo de instancia no tiene la misma arquitectura que la AMI especificada en la plantilla de lanzamiento.

Solución 1: no incluya tipos de instancias que tengan arquitecturas diferentes en su política de instancias mixtas.

1. Compruebe la arquitectura de la AMI mediante el comando describe-images o desde la consola Amazon EC2 comprobando el valor de la arquitectura en el panel de detalles de la página Amazon Machine Images (). AMIs
2. Compruebe la arquitectura de cada tipo de instancia que desee incluir en la política de instancias mixtas mediante el describe-instance-typescomando o desde la consola Amazon EC2 consultando la columna Arquitectura de la pantalla Tipos de instancias. Para obtener más información sobre

cómo se eligen tipos de instancia compatibles, consulte Compatibilidad para cambiar el tipo de instancia en la Guía del usuario de Amazon EC2.

3. Actualice o elimine los tipos de instancias incompatibles de su grupo de Auto Scaling mediante el update-auto-scaling-groupcomando.

Solución 2: para lanzar instancias ARM (Graviton2) y x86_64 (Intel) en el mismo grupo de escalado automático, debe usar plantillas de lanzamiento compatibles con una AMI compatible con ARM y una AMI compatible con Intel x86, respectivamente, para que coincidan con los tipos de instancias de su política de instancias mixtas.

1. Compruebe la arquitectura de la AMI en la plantilla de lanzamiento existente mediante el comando describe-images o desde la consola Amazon EC2 comprobando el valor de la arquitectura en el panel de detalles de la página Amazon Machine Images (). AMIs
2. Cree una nueva plantilla de lanzamiento mediante una AMI que coincida con la otra arquitectura que desee utilizar.
3. Actualiza tu grupo de Auto Scaling para anular la plantilla de lanzamiento existente y especifica la nueva plantilla de lanzamiento para cada tipo de instancia compatible mediante el update-auto-scaling-groupcomando. Para obtener más información, consulte Utilización de plantillas de lanzamiento múltiples.

La AMI “<ID de AMI>” está deshabilitada y no se puede ejecutar. El lanzamiento de la instancia EC2 ha producido un error.

Causa: está intentando lanzar instancias desde una AMI que se ha desactivado. Para obtener más información, consulte Deshabilitación de una AMI en la Guía del usuario de Amazon EC2.

Solución:

1. Cree una nueva plantilla de lanzamiento o configuración de lanzamiento y especifique una AMI que no esté deshabilitada.
2. Actualice el grupo de Auto Scaling con la nueva plantilla de lanzamiento o configuración de lanzamiento usando el comando update-auto-scaling-group.

Solución de problemas de Amazon EC2 Auto Scaling: problemas del equilibrador de carga

En esta página se proporciona información sobre los problemas causados por el balanceador de carga asociado con el grupo de Auto Scaling, las causas posibles y los pasos que puede realizar para resolver los problemas.

Para recuperar un mensaje de error, consulte Recuperación de un mensaje de error de las actividades de escalado.

Cuando no se pueden lanzar instancias EC2 debido a problemas con el balanceador de carga asociado al grupo de Auto Scaling, es posible que aparezcan uno o varios de los mensajes de error siguientes.

Problemas del balanceador de carga

- No se encontraron uno o varios grupos de destino. Error al validar la configuración del balanceador de carga.
- No se encuentra el equilibrador de carga <su equilibrador de carga>. Error al validar la configuración del balanceador de carga.
- No hay ningún balanceador de carga ACTIVO denominado <nombre del balanceador de carga>. Error al actualizar la configuración del balanceador de carga.
- La instancia EC2 <ID de instancia> no está en VPC. Error al actualizar la configuración del balanceador de carga.

> **Note**
>
> Puede usar el analizador de accesibilidad para solucionar problemas de conectividad. Para ello, compruebe si se puede acceder a las instancias de su grupo de escalado automático a través del equilibrador de carga. Para obtener más información sobre los distintos problemas de configuración de la red que el analizador de accesibilidad detecta automáticamente, consulte Reachability Analyzer explanation codes (Códigos de explicación del analizador de accesibilidad) en la Guía del usuario del analizador de accesibilidad.

No se encontraron uno o varios grupos de destino. Error al validar la configuración del balanceador de carga.

Problema: cuando el grupo de escalado automático lanza instancias, Amazon EC2 Auto Scaling intenta validar la existencia de los recursos de Elastic Load Balancing que están asociados al grupo de escalado automático. Cuando no se puede encontrar un grupo de destino, falla la actividad de escalado y se obtiene el error `One or more target groups not found. Validating load balancer configuration failed.`.

Causa 1: se ha eliminado un grupo de destino asociado al grupo de escalado automático.

Solución 1: puede crear un nuevo grupo de Auto Scaling sin el grupo de destino o eliminar el grupo de destino no utilizado del grupo de Auto Scaling mediante la consola Auto Scaling de Amazon EC2 o detach-load-balancer-targetel comando -groups.

Causa 2: el grupo de destino existe, pero se ha producido un problema al intentar especificar el ARN del grupo de destino al crear el grupo de escalado automático. Los recursos no se crean en el orden correcto.

Solución 2: cree un nuevo grupo de escalado automático y especifique el nombre del grupo de destino al final.

No se encuentra el equilibrador de carga <su equilibrador de carga>. Error al validar la configuración del balanceador de carga.

Problema: cuando el grupo de escalado automático lanza instancias, Amazon EC2 Auto Scaling intenta validar la existencia de los recursos de Elastic Load Balancing que están asociados al grupo de escalado automático. Cuando no se encuentra un equilibrador de carga clásico, falla la actividad de escalado y se obtiene el error `Cannot find Load Balancer <your load balancer>. Validating load balancer configuration failed.`.

Causa 1: el equilibrador de carga clásico se ha eliminado.

Solución 1: Puede crear un nuevo grupo de Auto Scaling sin el balanceador de cargas o eliminar el balanceador de cargas no utilizado del grupo Auto Scaling mediante la consola Auto Scaling de Amazon EC2 o el comando. detach-load-balancers

Causa 2: el Equilibrador de carga clásico existe, pero ha habido un problema al intentar especificar el nombre del equilibrador de carga al crear el grupo de escalado automático. Los recursos no se crean en el orden correcto.

Solución 2: Cree un nuevo grupo de Auto Scaling y especifique el nombre del balanceador de carga al final.

No hay ningún balanceador de carga ACTIVO denominado <nombre del balanceador de carga>. Error al actualizar la configuración del balanceador de carga.

Causa: es posible que se haya eliminado el balanceador de carga especificado.

Solución: Puede crear un nuevo balanceador de carga y después un nuevo grupo de Auto Scaling o puede crear un nuevo grupo de Auto Scaling sin el balanceador de carga.

La instancia EC2 <ID de instancia> no está en VPC. Error al actualizar la configuración del balanceador de carga.

Causa: la instancia especificada no existe en la VPC.

Solución: Puede eliminar el balanceador de carga asociado a la instancia o crear un nuevo grupo de Auto Scaling.

Solución de problemas de Amazon EC2 Auto Scaling: plantillas de lanzamiento

Utilice la información siguiente para diagnosticar y solucionar los problemas comunes que puedan surgir cuando trate de especificar una plantilla de lanzamiento del grupo de escalado automático.

No se pueden lanzar instancias

Si no puede lanzar ninguna instancia con una plantilla de lanzamiento ya especificada, verifique lo siguiente para la solución de problemas generales: Solución de problemas de Amazon EC2 Auto Scaling: errores de lanzamiento de instancias de EC2.

Debe usar una plantilla de lanzamiento válida y completa (valor no válido)

Problema: cuando intenta especificar una plantilla de lanzamiento para un grupo de escalado automático, obtiene el error `You must use a valid fully-formed launch template`. Puede que encuentre este error porque los valores de la plantilla de lanzamiento solo se validan cuando se crea o actualiza un grupo de escalado automático que la utiliza.

Causa 1: si recibe un error `You must use a valid fully-formed launch template`, hay problemas que hacen que Amazon EC2 Auto Scaling considere que la plantilla de lanzamiento no es válida. Se trata de un error genérico que puede tener varias causas diferentes.

Solución 1: pruebe los siguientes pasos para solucionar un error:

1. Preste atención a la segunda parte del mensaje de error para obtener más información. Tras el error `You must use a valid fully-formed launch template`, consulte el mensaje de error más específico que identifica el problema que tendrá que solucionar.
2. Si no puede encontrar la causa, pruebe la plantilla de lanzamiento con el comando run-instances. Use la opción `--dry-run`, como se muestra en el siguiente ejemplo. Esto le permite reproducir el problema y proporcionar información sobre su causa.

   ```
   aws ec2 run-instances --launch-template LaunchTemplateName=my-template,Version='1' --
   dry-run
   ```
3. Si un valor no es válido, verifique que el recurso especificado existe y que es correcto. Por ejemplo, cuando especifica un par de claves de Amazon EC2, el recurso debe existir en su cuenta y en la región en la que está creando o actualizando su grupo de escalado automático.
4. Si falta la información esperada, verifique la configuración y ajuste la plantilla de lanzamiento según sea necesario.
5. Después de realizar los cambios, vuelva a ejecutar el comando run-instances con la opción `--dry-run` para verificar que la plantilla de lanzamiento utiliza valores válidos.

Para obtener más información, consulte Creación de una plantilla de lanzamiento para un grupo de Auto Scaling.

No cuenta con autorización para utilizar la plantilla de lanzamiento (permisos insuficientes)

Problema: cuando intenta especificar una plantilla de lanzamiento para un grupo de escalado automático, obtiene el error `You are not authorized to use launch template`.

Causa 1: Si está intentando utilizar una plantilla de lanzamiento y no tiene suficientes credenciales de IAM, recibirá un error que indica que no está autorizado a utilizar la plantilla de lanzamiento.

Solución 1: para resolver este problema, pruebe lo siguiente:

- Verifique que las credenciales de IAM que está utilizando para realizar la solicitud tienen permisos para llamar a las acciones de la API de EC2 que necesita, incluida la acción `ec2:RunInstances`. Si especificó alguna etiqueta en la plantilla de lanzamiento, también debe tener permiso para usar la acción `ec2:CreateTags`.
- Como alternativa, verifique que las credenciales de IAM que está utilizando para realizar la solicitud tienen asignada la política `AmazonEC2FullAccess`. Esta política AWS gestionada otorga acceso total a todos los recursos y servicios relacionados de Amazon EC2, incluidos Amazon EC2 Auto Scaling y Elastic Load Balancing. CloudWatch

Para obtener más información sobre los permisos requeridos para usar plantillas de inicialización, incluidas políticas de IAM de ejemplo, consulte Controle el acceso a las plantillas de inicialización con permisos de IAM en la Guía del usuario de Amazon EC2. Para ver otros ejemplos de políticas de IAM, consulte Control del uso de la plantilla de inicialización de Amazon EC2 en los grupos de escalado automático.

Causa 2: Si está intentando utilizar una plantilla de lanzamiento que especifica un perfil de instancias, debe tener el permiso de IAM para pasar el rol de IAM asociado con el perfil de instancia.

Solución 2: verifique que las credenciales de IAM que está utilizando para hacer la solicitud tienen los permisos `iam:PassRole` adecuados para pasar el rol especificado al servicio de Amazon EC2 Auto Scaling. Para obtener más información y una política de IAM de ejemplo, consulte Rol de IAM para aplicaciones que se ejecuten en instancias de Amazon EC2. Para obtener más información sobre los temas relacionados con la solución de problemas, consulte Solución de problemas de Amazon EC2 e IAM en la Guía del usuario de IAM.

Causa 3: si intenta utilizar una plantilla de lanzamiento que especifica una AMI en otra Cuenta de AWS, y la AMI es privada y no se comparte con la Cuenta de AWS que está utilizando, recibirá un error que indica que no está autorizado a utilizar la plantilla de lanzamiento.

Solución 3: verifique que los permisos de la AMI incluyan la cuenta que está utilizando. Para obtener más información, consulte Compartir una AMI con Cuentas de AWS específicas en la Guía del usuario de Amazon EC2.

Información relacionada

Los recursos relacionados siguientes pueden serle de ayuda cuando trabaje con este servicio.

Recurso	Description (Descripción)
Referencia de API de Amazon EC2 Auto Scaling	La documentación de cada operación de la API muestra los parámetros de la solicitud y la respuesta de XML, y proporciona enlaces a temas de referencia del SDK específicos del idioma.
Auto Scaling en la Referencia de comando de AWS CLI	Descripciones de los AWS CLI comandos que puede usar para trabajar con grupos de Auto Scaling.
Herramientas de AWS para PowerShell Referencia de cmdlets	Las AWS herramientas le PowerShell permiten programar operaciones en sus AWS recursos desde la PowerShell línea de comandos.
Cree grupos de Auto Scaling con AWS CloudFormation	El recurso AWS::AutoScaling::AutoScalingAWS::AutoScaling::AutoScalingGroup le permite crear, modelar y administrar sus grupos de Auto Scaling sin acciones manuales.
Puntos de conexión y cuotas de Amazon EC2 Auto Scaling en Referencia general de AWS	Información acerca de las regiones y los puntos de conexión de Amazon EC2 Auto Scaling.
Página del producto	La página web principal con información sobre Amazon EC2 Auto Scaling.
AWS Re:post	AWS servicio gestionado de preguntas y respuestas (Q & A) que ofrece respuestas a sus preguntas técnicas revisadas por expertos y de colaboración colectiva.

Recurso	Description (Descripción)
Creación de una AMI en la Guía del usuario de Amazon EC2	Aprenda a crear una imagen de máquina de Amazon (AMI) a partir de una instancia personalizada.
Conectar a su instancia de Linux en la Guía del usuario de Amazon EC2	Aprenda a conectarse a las instancias de Linux que lance.
Conectar a su instancia de Windows en la Guía del usuario de Amazon EC2	Aprenda a conectarse a las instancias de Windows que lance.
Cómo crear una alarma de facturación para controlar tus AWS cargos estimados en la Guía del CloudWatch usuario de Amazon	Aprenda a controlar sus cargos estimados utilizando CloudWatch.
Guía del usuario de la aplicación Auto Scaling	Aprenda a configurar el escalado automático para recursos escalables para Amazon Web Services más allá de Amazon EC2.

Historial de revisión

En la siguiente tabla se describen los cambios importantes de la documentación de Amazon EC2 Auto Scaling, a partir de julio de 2018. Para obtener notificaciones sobre las actualizaciones de esta documentación, puede suscribirse a la fuente RSS.

Cambio	Descripción	Fecha
Distribución por zonas de disponibilidad: luego, se equilibran las reservas	Auto Scaling ahora ofrece una estrategia de distribución de reservas y zonas de disponibilidad equilibrada que lanza primero las instancias en sus reservas de capacidad y, a continuación, equilibra la capacidad restante entre las zonas de disponibilidad. Para obtener más información, consulte Distribución por zonas de disponibilidad en el grupo de escalado automático.	30 de junio de 2026
Actualización de seguridad de IAM	AutoScalingServiceRolePolicyAhora incluye el `ec2:CreateReplaceRootVolumeTask` permiso que permite reemplazar los volúmenes raíz en las instancias como parte de una actualización de la instancia. Además, el `events:DescribeRule` permiso se ha eliminado de la política.	20 de noviembre de 2025
Reemplace los volúmenes raíz	La sustitución del volumen raíz actualiza las instancias sustituyendo únicamente	20 de noviembre de 2025

	el volumen raíz de EBS y, al mismo tiempo, mantiene la instancia en ejecución. Para obtener más informaci ón, consulta Reemplazar los volúmenes raíz durante la actualización de la instancia.	
Políticas del ciclo de vida de las	Las políticas del ciclo de vida de las instancias le permiten controlar lo que ocurre cuando los bloqueos del ciclo de vida de las terminaciones no se completan correctamente. Para obtener más informaci ón, consulta Reemplazar los volúmenes raíz durante la actualización de la instancia.	20 de noviembre de 2025
Aprovisionamiento sincrónico	Ahora puede usar la LaunchInstances API para lanzar instancias de forma sincrónica con comentarios inmediatos y tener un control preciso sobre las instancia s de Availability Zone que se lanzan. Para obtener más información, consulte Aprovisionamiento sincrónico.	18 de noviembre de 2025

Destino de notificaciones de enlaces de ciclo de vida	Ahora puede dirigir las notificaciones de enlaces del ciclo de vida directamente a AWS Lambda las funciones. Para obtener más información, consulte Enrutar las notificaciones AWS Lambda directamente a.	28 de julio de 2025
High-resolution métricas	El seguimiento de Target ahora admite CloudWatch métricas de alta resolución con puntos de datos de segundo nivel que se publican a intervalos inferiores a un minuto. Para obtener más información, consulte Crear una política de seguimiento de objetivos con las métricas de alta resolución para obtener una respuesta más rápida.	22 de noviembre de 2024
Actualización de seguridad de IAM	La política AutoScalingServiceRolePolicygestionada ahora concede permisos adicionales a Resource Groups`resource-groups:ListGroupResources` .	20 de noviembre de 2024

Protección del rendimiento	Al utilizar la selección de tipo de instancia basada en atributos para su grupo de escalado automático, podrá habilitar la protección de rendimiento para garantizar que los tipos de instancia seleccionados sean equivalentes o superiores a un valor de referencia de rendimiento especificado. Para obtener más información, consulte Crear un grupo de instancias mezcladas con la selección de tipo de instancia basada en atributos.	20 de noviembre de 2024
Preferencia de reserva de capacidad	Ahora puede priorizar las instancias de lanzamiento en las reservas de capacidad. Para obtener información, consulte las Reservas de capacidad.	20 de noviembre de 2024
Cambio de zona	Ahora puede usar el cambio de zona para recuperar de las deficiencias de las aplicaciones en una zona de disponibilidad. Para obtener más información, consulte Cambio de zona en el grupo de escalado automático.	18 de noviembre de 2024

Distribución de zonas de disponibilidad	Ahora puede elegir una distribución de zona de disponibilidad para su grupo de escalado automático. Para obtener más informaci ón, consulte Distribución por zonas de disponibilidad en el grupo de escalado automático.	7 de noviembre de 2024
Actualización de seguridad de IAM	La política AutoScalingService RolePolicygestionada ahora concede permisos adicionales a Amazon EC2 (`ec2:GetSe curityGroupsForVpc` `yec2:GetInstanceTyp esFromInstanceRequ irements`).	29 de febrero de 2024
Además, se admite la hibernación en piscinas calientes Regiones de AWS	Ahora puedes hibernar las instancias de una piscina caliente en dos regiones adicionales: AWS GovCloud (US-East) y AWS GovCloud (US-West). Para obtener más información sobre grupos de calentamiento, consulte Warm pools for Amazon EC2 Auto Scaling (Grupos de calentami ento para Amazon EC2 Auto Scaling) en Amazon EC2 Auto Scaling User Guide (Guía del usuario de Amazon EC2 Auto Scaling).	26 de febrero de 2024

Además, se admite la hibernación en piscinas calientes Regiones de AWS	Ahora puede hibernar instancias en un grupo de calentamiento en dos regiones adicionales: Europa (Zúrich) y Oriente Medio (EAU). Para obtener más información sobre grupos de calentami ento, consulte Warm pools for Amazon EC2 Auto Scaling (Grupos de calentamiento para Amazon EC2 Auto Scaling) en Amazon EC2 Auto Scaling User Guide (Guía del usuario de Amazon EC2 Auto Scaling).	21 de febrero de 2024
Compatibilidad con el uso del parámetro entre cuentas	Ahora puede usar un AWS Systems Manager parámetro compartido desde otro Cuenta de AWS con Amazon EC2 Auto Scaling. Para obtener más información, consulte Uso de AWS Systems Manager parámetros en lugar de ID de AMI en las plantillas de lanzamiento en la Guía del usuario de Amazon EC2 Auto Scaling.	21 de febrero de 2024

Nueva opción de protección de precios de spot	Ahora puede definir su umbral de protección de precios para las instancias puntuales como un porcentaje del On-Demand precio al seleccionar el tipo de instancia en función de los atributos. Para obtener más información, consulte Protección de precios en la Guía del usuario de Amazon EC2 Auto Scaling.	29 de enero de 2024
Políticas de mantenimiento de instancias	Ahora puede usar una política de mantenimiento de instancias para definir si las instancias se lanzan antes o después de que se finalicen las instancias existentes durante los eventos que provocan su reemplazo, incluida una actualización de instancias. Para obtener más información, consulte Políticas de mantenimiento de instancias en la Guía del usuario de Amazon EC2 Auto Scaling.	15 de noviembre de 2023

Bloques de capacidad para ML	Ahora puede lanzar instancias en un bloque de capacidad especificando el ID de reserva del bloque de capacidad al crear una plantilla de lanzamiento. Con los bloques de capacidad, usted puede reservar instancias de GPU para el futuro a fin de respaldar sus cargas de trabajo de machine learning (ML) de corta duración. Para obtener más información, consulte Use Capacity Blocks for machine learning workloads en la Guía del usuario de Amazon EC2 Auto Scaling.	31 de octubre de 2023
Nuevas características de actualización de instancias	Ahora puede configurar la actualización de una instancia para establecer su estado como fallido y, si lo desea, revertirla cuando detecte que una CloudWatch alarma específica ha pasado a ese estado. `ALARM` Para obtener más información, consulte Inversión de cambios con una reversión en la Guía del usuario de Amazon EC2 Auto Scaling.	31 de julio de 2023

Cambios en la guía	Se ha agregado a la guía un tema nuevo sobre el lanzamiento de On-Demand instancias con reservas de capacidad. Para obtener más información, consulte Uso de reservas de On-Demand capacidad para reservar capacidad en zonas de disponibilidad específicas en la Guía del usuario de Amazon EC2 Auto Scaling.	28 de julio de 2023
Cambios en la guía	Se ha añadido a la guía un tema nuevo sobre la migración de las AWS CloudFormation pilas de configuraciones de lanzamiento a plantillas de lanzamiento. Para obtener más información, consulte Migre pilas de AWS CloudFormation de configuraciones de lanzamiento a plantillas de lanzamiento en la Guía del usuario de Amazon EC2 Auto Scaling.	18 de abril de 2023

Compatibilidad con nuevas operaciones de API	Esta versión añade tres nuevas operaciones de API: `AttachTrafficSources`, `DetachTrafficSources` y `DescribeTrafficSources`. Además, se ha agregado un nuevo campo, `TrafficSources`, a los resultados de las operaciones de `DescribeAutoScalingGroups`. Se ha agregado un nuevo estado de actividad, `WaitingForConnectionDraining`, a los resultados de las operaciones de `DescribeScalingActivities`. Amazon EC2 Auto Scaling también admite un nuevo valor, `VPC_LATTICE`, para el campo `HealthCheckType` en las operaciones de `CreateAutoScalingGroup`, `UpdateAutoScalingGroup` y `DescribeAutoScalingGroups`. Para obtener más información, consulte la Referencia de la API de Amazon EC2 Auto Scaling.	31 de marzo de 2023

Compatibilidad con Amazon VPC Lattice	Esta es la versión de disponibilidad general de la VPC Lattice para Amazon EC2 Auto Scaling. Para obtener más información, consulte Dirigir el tráfico a su grupo de escalado automático con un grupo de destinos de VPC Lattice en la Guía del usuario de Amazon EC2 Auto Scaling.	31 de marzo de 2023
Cambios en la guía	La sección con AWS CLI ejemplos para trabajar con Elastic Load Balancing ahora incluye ejemplos nuevos y actualizados. Para obtener más información, consulte Ejemplos de cómo trabajar con Elastic Load Balancing with the AWS Command Line Interface (AWS CLI) en la Guía del usuario de Amazon EC2 Auto Scaling.	31 de marzo de 2023
Support para el escalado predictivo en forma adicional Regiones de AWS	Ahora puede crear políticas de escalado predictivo en las regiones de Oriente Medio (EAU) y AWS GovCloud (US-East). Para obtener más información, consulte Escalado predictivo para un grupo de Amazon EC2 Auto Scaling en la Guía del usuario de Amazon EC2 Auto Scaling.	16 de marzo de 2023

Nuevas características de actualización de instancias	Ahora puede optar por finalizar o ignorar las instancias en espera y reemplazar o ignorar las instancias protegidas contra la reducción horizontal, en lugar de esperar a que sean reemplazables. También puede revertir los cambios de una actualización de instancias con errores. Como parte de esta actualización, la documentación se amplió para incluir temas sobre cómo revertir una actualización de instancias, cancelar una actualización de instancias y comprender los valores predeterminados de los parámetros configurables de una actualización de instancias. Para obtener más información, consulte Sustitución de instancias de Auto Scaling en función de una actualización de instancias en la Guía del usuario de Amazon EC2 Auto Scaling.	10 de febrero de 2023

Support para usar un AWS Systems Manager parámetro para un ID de AMI	Ahora puede usar un parámetro Systems Manager en lugar de un ID de AMI en la plantilla de lanzamiento. Para obtener más informaci ón, consulte Usar parámetro s de AWS Systems Manager en lugar de ID de AMI en las plantillas de lanzamiento en la Guía del usuario de Amazon EC2 Auto Scaling.	19 de enero de 2023
Recomendaciones de escalado predictivo	Ahora puede obtener recomendaciones para evaluar y elegir la política de escalado predictivo desde la consola de Amazon EC2 Auto Scaling. Para obtener más información, consulte Evaluar las políticas de escalado predictivo en la Guía del usuario de Amazon EC2 Auto Scaling.	18 de enero de 2023
Pronósticos de escala predictiva	Las previsiones generadas por el escalado predictivo ahora se actualizan cada seis horas en vez de diariamente. Para obtener más información, consulte Escalado predictiv o para un grupo de Amazon EC2 Auto Scaling en la Guía del usuario de Amazon EC2 Auto Scaling.	6 de enero de 2023

Support para matemáticas CloudWatch métricas	Ahora puede utilizar la calculadora de métricas al crear políticas de escalado de seguimiento de destino. Con la matemática métrica, puedes consultar múltiples CloudWatch métricas y usar expresiones matemáticas para crear nuevas series temporales basadas en estas métricas. Para obtener más información, consulte Create a target tracking scaling policy for Amazon EC2 Auto Scaling using metric math (Creación de una política de Amazon EC2 Auto Scaling mediante una calculadora de métricas) en la Guía del usuario de Amazon EC2 Auto Scaling.	8 de diciembre de 2022
Actualización de permisos de roles vinculados a servicios de IAM	La política `AutoScalingServiceRolePolicy` ahora otorga permisos adicionales a Amazon EC2 Auto Scaling. Para obtener más información, consulte Políticas administradas por AWS para Amazon EC2 Auto Scaling en la Guía del usuario de Amazon EC2 Auto Scaling.	6 de diciembre de 2022

Nueva estrategia de asignación de instancias de spot	Ahora puede utilizar la estrategia de asignación optimizada por precio y capacidad para solicitar instancias de spot a los grupos de spot que tienen menos probabilidades de interrumpirse y tienen el precio más bajo posible. Para obtener más información, consulte Allocation strategies (Estrategias de asignación) en la Guía del usuario de Amazon EC2 Auto Scaling.	10 de noviembre de 2022
Support para la reducción horizontal predictiva en la región de Asia-Pacífico (Yakarta)	Ahora puede crear políticas de escalado predictivo en Asia Pacífico (Yakarta). Para obtener más información, consulte Escalado predictivo para un grupo de Amazon EC2 Auto Scaling en la Guía del usuario de Amazon EC2 Auto Scaling.	13 de octubre de 2022
Soporte para métricas personalizadas para reducción horizontal predictiva en la consola	Ahora puede usar métricas personalizadas al crear políticas de escalado predictivo desde la consola de Amazon EC2 Auto Scaling. Para obtener más información, consulte Escalado predictivo para un grupo de Amazon EC2 Auto Scaling en la Guía del usuario de Amazon EC2 Auto Scaling.	13 de octubre de 2022

CloudWatch supervisión de las métricas de escalamiento predictivo	Ahora puede acceder a los datos de monitoreo para utilizar el escalado predictivo CloudWatch. Esto le permite usar la matemática métrica para crear nuevas series temporales que muestren la precisión de los datos de las previsiones. Para obtener más información, consulte Supervisar las métricas de escalado predictivo CloudWatch en la Guía del usuario de Amazon EC2 Auto Scaling.	7 de julio de 2022
Soporte para reducción horizontal predictiva en la región de Asia-Pacífico (Osaka)	Ahora puede crear políticas de escalado predictivo en Asia Pacífico (Osaka). Para obtener más información, consulte Escalado predictivo para un grupo de Amazon EC2 Auto Scaling en la Guía del usuario de Amazon EC2 Auto Scaling.	6 de julio de 2022

Warm pool hibernation supported in additional Regions (Se admite la hibernación de grupos de calentamiento en regiones adicionales)	Ahora puede hibernar instancias en un grupo de calentamiento en cuatro regiones adicionales: África (Ciudad del Cabo), Asia-Pacífico (Yakarta), Asia-Pacífico (Osaka) y Europa (Milán). Para obtener más información sobre grupos de calentamiento, consulte Warm pools for Amazon EC2 Auto Scaling (Grupos de calentamiento para Amazon EC2 Auto Scaling) en Amazon EC2 Auto Scaling User Guide (Guía del usuario de Amazon EC2 Auto Scaling).	5 de julio de 2022
Actualización de comprobaciones de estado	Al realizar comprobaciones de estado, Amazon EC2 Auto Scaling ahora permite minimizar cualquier tiempo de inactividad que pueda producirse debido a problemas temporales o comprobaciones de estado mal configuradas. Para obtener más información, consulte Cómo Amazon EC2 Auto Scaling minimiza el tiempo de inactividad en la Guía del usuario de Amazon EC2 Auto Scaling.	21 de mayo de 2022

Preparación predeterminada de instancias	Ahora, para unificar todas las configuraciones de preparación y recuperación de un grupo de escalado automático y optimizar el rendimiento de las políticas de escalado que se escalan continuamente, puede habilitar la preparación predeterminada de instancias. Para obtener más información, consulte Establecer la preparación predeterminada de instancias para un grupo de escalado automático en la Guía del usuario de Amazon EC2 Auto Scaling.	19 de abril de 2022
Cambios en la guía	Se ha agregado a la guía un nuevo capítulo sobre la integración con otros AWS servicios. Para obtener más información, consulte Servicios de AWS integrados en Amazon EC2 Auto Scaling en la Guía del usuario de Amazon EC2 Auto Scaling.	29 de marzo de 2022

Actualización de permisos de roles vinculados a servicios de IAM	La política `AutoScalingServiceRolePolicy` ahora otorga permisos de lectura adicionales a Amazon EC2 Auto Scaling. Para obtener más información, consulte Políticas administradas por AWS para Amazon EC2 Auto Scaling en la Guía del usuario de Amazon EC2 Auto Scaling.	28 de marzo de 2022
Los metadatos de instancia proporcionan el estado de ciclo de vida de destino	Puede recuperar el estado de ciclo de vida de destino de una instancia de Auto Scaling a partir de los metadatos de instancia. Para obtener más información, consulte Recuperar el estado de ciclo de vida de destino a través de los metadatos de instancia en la Guía del usuario de Amazon EC2 Auto Scaling.	24 de marzo de 2022

Compatibilidad con la nueva funcionalidad de grupo de calentamiento	Ahora puede hibernar instancias en un grupo de calentamiento para detener instancias sin eliminar su contenido de memoria (RAM). Ahora también puede devolver instancias al grupo de calentamiento durante la reducción horizontal, en lugar de terminar siempre la capacidad de instancia que necesitará más adelante. Para obtener más información, consulte Grupos de calentamiento para Amazon EC2 Auto Scaling en la Guía del usuario de Amazon EC2 Auto Scaling.	24 de febrero de 2022
Cambios en la guía	La consola de Amazon EC2 Auto Scaling se actualizó con opciones adicionales que lo ayudarán a iniciar una actualización de instancias con la omisión de coincidencias habilitada y una configuración deseada especificada. Para obtener más información, consulte Inicio o cancelación de una actualización de instancias (consola) en la Guía del usuario de Amazon EC2 Auto Scaling.	3 de febrero de 2022

Métricas personalizadas para políticas de escalado predictivo	Ahora puede elegir si desea utilizar métricas personalizadas cuando cree políticas de escalado predictivo. También puede utilizar cálculos métricos para personalizar aún más las métricas que se incluyan en la política. Para obtener más información, consulte Advanced predictive scaling policy configurations using custom metrics (Configuraciones avanzadas de políticas de escalado predictivo mediante métricas personalizadas).	24 de noviembre de 2021
Nueva estrategia On-Demand de asignación	Ahora puede elegir si desea lanzar On-Demand instancias en función del precio (primero los tipos de instancias con el precio más bajo) al crear un grupo de Auto Scaling que usa una política de instancias mixtas. Para obtener más información, consulte Allocation strategies (Estrategias de asignación) en la Guía del usuario de Amazon EC2 Auto Scaling.	27 de octubre de 2021

Attribute-based selección del tipo de instancia	Amazon EC2 Auto Scaling agrega compatibilidad con la selección del tipo de instancia basada en atributos. En lugar de elegir los tipos de instancia de forma manual, puede expresar sus requisitos de instancia como un conjunto de atributos, como vCPU, memoria y almacenamiento. Para obtener más informaci ón, consulte Creating an Auto Scaling group using attribute-based instance type selection (Creación de un grupo de escalado automátic o mediante la selección del tipo de instancia basada en atributos) en la Guía del usuario de Amazon EC2 Auto Scaling.	27 de octubre de 2021

Compatibilidad con el filtrado de grupos por etiquetas	A partir de ahora se pueden filtrar los grupos de Auto Scaling mediante filtros de etiquetas cuando se recupere la información sobre los grupos de Auto Scaling con el comando `describe-auto-scaling-groups` . Para obtener más información, consulte Use tags to filter Auto Scaling groups (Uso de etiquetas para filtrar grupos de Auto Scaling) en la Guía del usuario de Amazon EC2 Auto Scaling.	14 de octubre de 2021
Cambios en la guía	Se ha actualizado la consola Amazon EC2 Auto Scaling para ayudarle a crear políticas AWS Lambda de rescisión personalizadas con. También se ha revisado la documenta ción de la consola. Para obtener más informaci ón, consulte Utilización de diferentes políticas de terminación (consola).	14 de octubre de 2021
Compatibilidad con la copia de configuraciones de inicio en plantillas de lanzamiento	Ahora puede copiar todas las configuraciones de lanzamien to de una AWS región en nuevas plantillas de lanzamien to desde la consola Amazon EC2 Auto Scaling.	9 de agosto de 2021

Expande la funcionalidad de actualización de instancias	Ahora puede incluir actualizaciones, como una nueva versión de una plantilla de lanzamiento, al reemplazar instancias agregando la configuración deseada al comando `start-instance-refresh` . También puede omitir la sustitución de instancias que ya tienen la configuración deseada habilitando la coincidencia de omisiones. Para obtener más información, consulte Sustitución de instancias de Auto Scaling en función de una actualización de instancias en la Guía del usuario de Amazon EC2 Auto Scaling.	5 de agosto de 2021
Compatibilidad con políticas de terminación personalizadas	Ahora puede crear políticas de terminación personalizadas con AWS Lambda. Para obtener más información, consulte Creación de una política de terminación personalizada con Lambda. La documentación para especificar las políticas de terminación se ha actualizado en consecuencia.	29 de julio de 2021

Cambios en la guía	La consola de Amazon EC2 Auto Scaling se ha actualizado y mejorado con características adicionales que le ayudarán a crear acciones programadas con una zona horaria especific ada. La documentación para el Escalado programado se ha revisado en consecuencia.	3 de junio de 2021
Volúmenes gp3 en configura ciones de lanzamiento	Ahora puede especific ar volúmenes gp3 en las asignaciones de dispositivo de bloques para configuraciones de lanzamiento.	2 de junio de 2021
Compatibilidad con el escalado predictivo	Ahora puede utilizar el escalado predictivo para escalar de forma proactiva los grupos de Amazon EC2 Auto Scaling mediante una política de escalado. Para obtener más información, consulte Escalado predictivo para un grupo de Amazon EC2 Auto Scaling en la Guía del usuario de Amazon EC2 Auto Scaling. Con esta actualización, la política AutoScalingService RolePolicygestionada ahora incluye el permiso para activar la acción de la `cloudwatc h:GetMetricData` API.	19 de mayo de 2021

Cambios en la guía	Ahora puedes acceder a plantillas de ejemplo para enlaces sobre el ciclo de vida desde GitHub. Para obtener más información, consulte Enlaces de ciclo de vida de Amazon EC2 Auto Scaling en la guía del usuario de Amazon EC2 Auto Scaling.	9 de abril de 2021
Compatibilidad con grupos de calentamiento	Ahora puede equilibrar el rendimiento (minimizar los inicios en frío) y el costo (detener el aprovisionamiento excesivo de la capacidad de instancia) para aplicaciones con largos tiempos de primer arranque agregando grupos de calentamiento a grupos de Auto Scaling. Para obtener más información, consulte Grupos de calentamiento para Amazon EC2 Auto Scaling en la Guía del usuario de Auto Scaling de Amazon EC2.	8 de abril de 2021

Compatibilidad con los puntos de control	Ahora puede agregar puntos de control a una actualización de instancia para reemplazar instancias en fases y realizar verificaciones en las instancias en puntos específicos. Para obtener más información, consulte Agregar puntos de control a una actualización de instancia en la Guía del usuario de Amazon EC2 Auto Scaling.	18 de marzo de 2021
Cambios en la guía	Documentación mejorada para su uso EventBridge con los eventos de Amazon EC2 Auto Scaling y los enlaces de ciclo de vida. Para obtener más información, consulte Uso de Amazon EC2 Auto Scaling EventBridge con un tutorial sobre cómo configurar un enlace de ciclo de vida que invoque una función de Lambda en la Guía del usuario de Amazon EC2 Auto Scaling.	18 de marzo de 2021

Compatibilidad para zonas horarias locales	Ahora puede crear acciones programadas recurrentes en la zona horaria local agregando la opción `--time-zone` para el comando put-scheduled-update-group-action. Si la zona horaria observa el horario de verano (DST), la acción recurrente se ajusta automáticamente para el horario de verano. Para obtener más información, consulte la sección sobre Escalado programado en la Guía del usuario de Amazon EC2 Auto Scaling.	9 de marzo de 2021
Amplía la funcionalidad de las políticas de instancias mixtas	Ahora puede asignar prioridad a los tipos de instancias para la capacidad de spot cuando utilice una política de instancias mixtas. Amazon EC2 Auto Scaling intenta cumplir con las prioridades sobre la base del mejor esfuerzo, pero optimiza primero la capacidad. Para obtener más información, consulte la sección sobre Grupos de escalado automático con varios tipos de instancia y opciones de compra en la guía del usuario de Amazon EC2 Auto Scaling.	8 de marzo de 2021

Actividades de escalado para grupos eliminados	Ahora puede ver las actividades de escalado para los grupos de Auto Scaling eliminados agregando la opción `--include-deleted-groups` del comando describe-scaling-activities. Para obtener más información, consulte Solución de problemas para Amazon EC2 Auto Scaling en la guía del usuario de Amazon EC2 Auto Scaling.	23 de febrero de 2021
Mejoras en la consola	Ahora puede crear y adjuntar un Application Load Balancer o un Network Load Balancer desde la consola de Amazon EC2 Auto Scaling. Para obtener más información, consulte Crear y adjuntar un Application Load Balancer o un Network Load Balancer (consola) en la guía del usuario de Amazon EC2 Auto Scaling.	24 de noviembre de 2020
Múltiples interfaces de red	Ahora puede configurar una plantilla de lanzamiento para un grupo de escalado automático que especifique varias interfaces de red. Para obtener más información, consulte Interfaces de red en la VPC.	23 de noviembre de 2020

Plantillas de lanzamiento múltiples	Ahora se pueden utilizar varias plantillas de lanzamiento con grupos de Auto Scaling. Para obtener más información, consulte Especificación de una plantilla de lanzamiento diferente para un tipo de instancia en la Guía del usuario de Amazon EC2 Auto Scaling.	19 de noviembre de 2020
Balanceadores de carga de gateway	Guía actualizada para mostrar cómo adjuntar un balanceador de carga de gateway a un grupo de escalado automático para garantizar que las instancias de dispositivo lanzadas por Amazon EC2 Auto Scaling se registren automáticamente y anulen el registro del balanceador de carga. Para obtener más información, consulte Tipos de Elastic Load Balancing y Adjuntar un balanceador de carga a al grupo de escalado automático en la guía del usuario de Amazon EC2 Auto Scaling.	10 de noviembre de 2020

Duración máxima de la instancia	Ahora puede reducir la duración máxima de la instancia a un día (86 400 segundos). Para obtener más información, consulte Sustitución de instancias de Auto Scaling en función de la duración máxima de las instancias en la guía del usuario de Amazon EC2 Auto Scaling.	9 de noviembre de 2020
Reequilibrio de la capacidad	Puede configurar su grupo de escalado automático para lanzar una instancia de spot de reemplazo cuando Amazon EC2 emita una recomendación de reequilibrio. Para obtener más información, consulte Reequilibrio de capacidad de Amazon EC2 Auto Scaling en la Guía del usuario de Amazon EC2 Auto Scaling.	4 de noviembre de 2020

Servicios de metadatos de instancia versión 2	Puede pedir el uso del Servicio de metadatos de instancia versión 2, que es un método orientado a la sesión para solicitar metadatos de instancia, cuando utilice configuraciones de lanzamiento. Para obtener más información, consulte Configuración de las opciones de metadatos de instancias en la Guía del usuario de Amazon EC2 Auto Scaling.	28 de julio de 2020
Cambios en la guía	Varias mejoras y nuevos procedimientos de consola en las secciones Controlar las instancias que Auto Scaling termina durante la reducción horizontal, Monitoreo de instancias y grupos de Auto Scaling, Plantillas de lanzamiento, y Configuraciones de lanzamiento de la guía del usuario de Amazon EC2 Auto Scaling.	28 de julio de 2020

Actualización de instancias	Comience una actualización de instancias para actualizar todas las instancias del grupo de escalado automático cuando realice un cambio de configuración. Para obtener más información, consulte Sustitución de instancias de Auto Scaling en función de una actualización de instancias en la Guía del usuario de Amazon EC2 Auto Scaling.	16 de junio de 2020
Cambios en la guía	Se han realizado varias mejoras en las secciones Sustitución de instancias de Auto Scaling en función de la duración máxima de la instancia, Grupos de escalado automático con varios tipos de instancia y opciones de compra, Escalado basado en Amazon SQS, y Etiquetado de grupos e instancias de la guía del usuario de Amazon EC2 Auto Scaling.	6 de mayo de 2020

Cambios en la guía	Varias mejoras en la documentación de IAM. Para obtener más información, consulte Compatibilidad con las plantillas de lanzamiento y Ejemplos de políticas basadas en identidades de Amazon EC2 Auto Scaling en la guía del usuario de Amazon EC2 Auto Scaling.	4 de marzo de 2020
Desactivación de las políticas de escalado	Ahora puede deshabilitar y volver a habilitar las políticas de escalado. Esta caracterí stica le permite deshabilitar temporalmente una política de escalado mientras conserva los detalles de configura ción para que pueda volver a habilitar la política más adelante. Para obtener más información, consulte Desactivación de una política de escalado para un grupo de escalado automático en la Guía del usuario de Amazon EC2 Auto Scaling.	18 de febrero de 2020

Adición de la funcionalidad de notificaciones	Amazon EC2 Auto Scaling ahora le envía eventos cuando Panel de AWS Health sus grupos de Auto Scaling no pueden ampliarse porque falta un grupo de seguridad o una plantilla de lanzamiento. Para obtener más informaci ón, consulte Notificaciones de Panel de estado para Amazon EC2 Auto Scaling en la guía del usuario de Amazon EC2 Auto Scaling.	12 de febrero de 2020
Cambios en la guía	Varias mejoras y correccio nes en las secciones Cómo funciona Amazon EC2 Auto Scaling con IAM, Ejemplos de políticas basadas en identidad es de Amazon EC2 Auto Scaling, Política de claves CMK requerida para su uso con volúmenes cifrados, y Monitoreo de instancias y grupos de Auto Scaling de la guía del usuario de Amazon EC2 Auto Scaling.	10 de febrero de 2020

Cambios en la guía	Documentación mejorada para grupos de Auto Scaling que utilizan la ponderación de instancias. Aprenda a utilizar las políticas de escalado cuando utilice “unidades de capacidad” para medir la capacidad deseada. Para obtener más información, consulte Cómo funcionan las políticas de escalado y Tipos de ajuste de escalado en la guía del usuario de Amazon EC2 Auto Scaling.	6 de febrero de 2020
Nuevo capítulo sobre seguridad	El nuevo capítulo Seguridad de la Guía del usuario de Amazon EC2 Auto Scaling lo ayuda a entender cómo aplicar el modelo de responsabilidad compartida cuando se utiliza Amazon EC2 Auto Scaling. En esta actualización, el capítulo "Control del acceso a los recursos de Amazon EC2 Auto Scaling" de la guía de usuario se ha sustituido por una sección nueva y más útil, Identity and Access Management para Amazon EC2 Auto Scaling.	4 de febrero de 2020

Recomendaciones para tipos de instancia	AWS Compute Optimizer proporciona recomendaciones de instancias de Amazon EC2 para ayudarle a mejorar el rendimiento, ahorrar dinero o ambas cosas. Para obtener más información, consulte Obtención de recomendaciones para un tipo de instancia en la guía del usuario de Amazon EC2 Auto Scaling.	3 de diciembre de 2019
Hosts dedicados y grupos de recursos de host	Guía actualizada para mostrar cómo crear una plantilla de lanzamiento que especifica un grupo de recursos del host. Esto le permite crear un grupo de escalado automático con una plantilla de lanzamiento que especifica una AMI BYOL para usar en hosts dedicados. 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.	3 de diciembre de 2019

Compatibilidad con puntos de conexión de Amazon VPC	Ahora puede establecer una conexión privada entre su VPC y Amazon EC2 Auto Scaling. Para obtener más información, consulte Amazon EC2 Auto Scaling y puntos de enlace de la VPC en la guía del usuario de Amazon EC2 Auto Scaling.	22 de noviembre de 2019
Duración máxima de la instancia	Ahora puede reemplazar instancias automáticamente especificando la duración máxima que una instancia puede estar en servicio. Si las instancias se acercan a este límite, Amazon EC2 Auto Scaling las va reemplazando gradualmente. Para obtener más información, consulte Sustitución de instancias de Auto Scaling en función de la duración máxima de las instancias en la guía del usuario de Amazon EC2 Auto Scaling.	19 de noviembre de 2019

Ponderación de instancias	En los grupos de Auto Scaling con varios tipos de instancia , si lo desea, ahora puede especificar el número de unidades de capacidad que cada tipo de instancia aporta a la capacidad del grupo. Para obtener más informaci ón, consulte Ponderación de instancias para Amazon EC2 Auto Scaling en la guía del usuario de Auto Scaling de Amazon EC2.	19 de noviembre de 2019
Número mínimo de tipos de instancia	Ya no es necesario especificar tipos de instancias adicionales para los grupos de instancias puntuales y reservadas. On-Demand En todos los grupos de Auto Scaling, el mínimo es ahora un tipo de instancia . Para obtener más informaci ón, consulte la sección sobre Grupos de escalado automátic o con varios tipos de instancia y opciones de compra en la guía del usuario de Amazon EC2 Auto Scaling.	16 de septiembre de 2019

Compatibilidad con una nueva estrategia de asignación de instancias de spot	Amazon EC2 Auto Scaling ahora es compatible con una nueva estrategia de asignación de instancias de spot "optimizada para la capacidad " que atiende la solicitud mediante grupos de instancias de spot que se eligen de forma óptima en función de la capacidad de spot disponible. Para obtener más información, consulte la sección sobre Grupos de escalado automático con varios tipos de instancia y opciones de compra en la guía del usuario de Amazon EC2 Auto Scaling.	12 de agosto de 2019
Cambios en la guía	Se mejoró la documentación de Amazon EC2 Auto Scaling en Service-linked los temas de roles y política de claves CMK requeridas para su uso con volúmenes cifrados.	1 de agosto de 2019
Compatibilidad con las mejoras de etiquetado	Ahora, Amazon EC2 Auto Scaling agrega etiquetas a las instancias de Amazon EC2 durante la misma llamada a la API que lanza las instancias. Para obtener más información, consulte Etiquetado de grupos e instancias de Auto Scaling.	26 de julio de 2019

Cambios en la guía	Se ha mejorado la documentación de Amazon EC2 Auto Scaling del tema Suspender y reanudar procesos de escalado. Se han actualizado los ejemplos de políticas administradas por el cliente para incluir una política de ejemplo que permite a los usuarios transferir a Amazon EC2 Auto Scaling únicamente los roles vinculados a servicios con un sufijo personalizado específico.	13 de junio de 2019
Compatibilidad de la nueva característica de Amazon EBS	Se ha añadido la compatibilidad de la nueva característica de Amazon EBS en el tema de la plantilla de lanzamiento. Cambie el estado de cifrado de un volumen de EBS al restaurarlo a partir de una instantánea. 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.	13 de mayo de 2019

Cambios en la guía	La documentación de Amazon EC2 Auto Scaling se ha mejorado en las siguiente s secciones: Control de las instancias que Auto Scaling termina durante la reducción horizontal, Grupos de Auto Scaling, Grupos de Auto Scaling con varios tipos de instancias y opciones de compra, y Escalado dinámico para Amazon EC2 Auto Scaling.	12 de marzo de 2019
Compatibilidad con la combinación de tipos de instancia y opciones de compra	Aprovisione y escale automáticamente las instancia s entre las opciones de compra (instancias puntuales e instancias reservadas) y los tipos de instancias dentro de un único grupo de Auto Scaling. On-Demand Para obtener más información, consulte la sección sobre Grupos de escalado automátic o con varios tipos de instancia y opciones de compra en la guía del usuario de Amazon EC2 Auto Scaling.	13 de noviembre de 2018

Tema actualizado para escalar basado en Amazon SQS	Se ha actualizado la guía para explicar cómo puede utilizar métricas personalizadas para escalar un grupo de escalado automático en respuesta a los cambios en la demanda de una cola de Amazon SQS. Para obtener más información, consulte Escalado en función de Amazon SQS en la guía del usuario de Amazon EC2 Auto Scaling.	26 de julio de 2018

En la siguiente tabla se describen cambios importantes en la documentación de Amazon EC2 Auto Scaling antes de julio de 2018.

Característica	Description (Descripción)	Fecha de lanzamiento de la nueva versión
Compatibilidad con las políticas de escalado de seguimiento de destino	Configure el escalado dinámico de la aplicación en unos pocos pasos. Para obtener más información, consulte Políticas de escalado de seguimiento de destino de Amazon EC2 Auto Scaling.	12 de julio de 2017
Compatibilidad con los permisos de nivel de recursos	Cree políticas de IAM para controlar el acceso en el nivel de recursos. Para obtener más información, consulte Control del acceso a los recursos de Amazon EC2 Auto Scaling.	15 de mayo de 2017
Mejoras en la monitorización	Las métricas de los grupos de Auto Scaling ya no necesitan que se habilite el monitoreo detallado. Ahora puede habilitar la recopilación de métricas del grupo y ver los gráficos de las métricas en la pestaña Monitorin g de la consola. Para obtener más información, consulte	18 de agosto de 2016

Característica	Description (Descripción)	Fecha de lanzamiento de la nueva versión
	Supervisión de sus grupos e instancias de Auto Scaling mediante Amazon CloudWatch.	
Compatibilidad con Application Load Balancers	Adjunte uno o varios grupos de destino a un grupo de escalado automático nuevo o existente. Para obtener más información, consulte Adjuntar un balanceador de carga a su grupo de escalado automático.	11 de agosto de 2016
Eventos de enlaces de ciclo de vida	Amazon EC2 Auto Scaling envía los eventos EventBridge a cuando llama a los enlaces del ciclo de vida. Para obtener más información, consulte Cómo obtener EventBridge cuándo escala su grupo de Auto Scaling.	24 de febrero de 2016
Protección de instancias	Impida que Amazon EC2 Auto Scaling; seleccione instancias específicas para su terminación al reducir horizontalmente. Para obtener más información, consulte Protección de instancias.	07 de diciembre de 2015
Políticas de escalado por pasos	Cree una política de escalado que le permita escalar en función del tamaño de la interrupción de alarma. Para obtener más información, consulte Tipos de políticas de escalado.	06 de julio de 2015
Actualizar el balanceador de carga	Adjunte un balanceador de carga o desconecte un balanceador de carga de un grupo de escalado automático existente. Para obtener más información, consulte Adjuntar un balanceador de carga a su grupo de escalado automático.	11 de junio de 2015

Característica	Description (Descripción)	Fecha de lanzamiento de la nueva versión
Support para ClassicLink	Vincule EC2-Classic las instancias de su grupo de Auto Scaling a una VPC para permitir la comunicación entre estas EC2-Classic instancias vinculadas y las instancia s de la VPC mediante direcciones IP privadas. Para obtener más información, consulte Vincular EC2-Classic instancias a una VPC.	19 de enero de 2015
Enlaces de ciclo de vida	Mantenga sus instancias recién lanzadas o terminada s en un estado pendiente mientras realiza acciones en ellas. Para obtener más información, consulte Enlaces de ciclo de vida de Amazon EC2 Auto Scaling.	30 de julio de 2014
Desasociar instancias	Desconecte instancias de un grupo de escalado automático. Para obtener más información, consulte Separar las instancias EC2 del grupo de escalado automático.	30 de julio de 2014
Poner las instancias en estado de espera	Ponga las instancias que se encuentran en estado `InService` en el estado `Standby`. Para obtener más información, consulte Eliminar temporalmente instancias de su grupo de escalado automático.	30 de julio de 2014
Administración de etiquetas	Administre los grupos de Auto Scaling con la Consola de administración de AWS. Para obtener más informaci ón, consulte Etiquetado de grupos e instancias de Auto Scaling.	01 de mayo de 2014
Compatibilidad con instancias dedicadas	Lance instancias dedicadas especificando un atributo de tenencia de ubicación cuando cree una configuración de lanzamiento. Para obtener más información, consulte Tenencia de ubicación de instancias.	23 de abril de 2014

Característica	Description (Descripción)	Fecha de lanzamiento de la nueva versión
Crear un grupo o configuración de lanzamiento a partir una instancia EC2	Cree un grupo de escalado automático o una configuración de lanzamiento mediante una instancia EC2. Para obtener información sobre cómo crear una configuración de lanzamiento mediante una instancia EC2, consulte Crear una configuración de lanzamiento con una instancia EC2. Para obtener más información acerca de cómo crear un grupo de escalado automático mediante una instancia EC2, consulte Creación de un grupo de escalado automático mediante una instancia EC2.	02 de enero de 2014
Asociar instancias	Habilite el escalado automático para una instancia EC2, asociando la instancia a un grupo de escalado automático existente. Para obtener más información, consulte Adjuntar instancias EC2 a su grupo de escalado automático.	02 de enero de 2014
Ver límites de la cuenta	Consulte los límites de los recursos de Auto Scaling para su cuenta. Para obtener más información, consulte Cuotas para los recursos y grupos de escalado automático.	02 de enero de 2014
Compatibilidad con la consola para Amazon EC2 Auto Scaling	Acceda a Amazon EC2 Auto Scaling mediante Consola de administración de AWS. Para obtener más información, consulte Introducción a Amazon EC2 Auto Scaling.	10 de diciembre de 2013
Asignar una dirección IP pública	Asigne una dirección IP pública a una instancia lanzada en una VPC. Para obtener más información, consulte Lanzar instancias de Auto Scaling en una VPC.	19 de septiembre de 2013

Característica	Description (Descripción)	Fecha de lanzamiento de la nueva versión
Política de terminación de instancias	Especifique una política de terminación de instancias que use Amazon EC2 Auto Scaling cuando se terminen las instancias EC2. Para obtener más información, consulte Controlar las instancias que Auto Scaling termina durante la reducción horizontal.	17 de septiembre de 2012
Compatibilidad con los roles de IAM	Lance instancias EC2 con un perfil de instancias de IAM. Puede utilizar esta característica para asignar roles de IAM a sus instancias, lo que permite que las aplicaciones tengan acceso a otros servicios de Amazon Web Services de forma segura. Para obtener más información, consulte Lanzar instancias de Auto Scaling; con un rol de IAM.	11 de junio de 2012
Compatibilidad con instancias de spot	Lance las instancias de spot con una configuración de lanzamiento. Para obtener más información, consulte Requesting Spot Instances for fault-tolerant and flexible applications (Solicitud de instancias de spot para aplicaciones flexibles y tolerantes a errores).	7 de junio de 2012
Etiquetar grupos e instancias	Etiquete grupos de Auto Scaling y especifique que la etiqueta se aplique también a las instancias EC2 lanzadas después de la creación de la etiqueta. Para obtener más información, consulte Etiquetado de grupos e instancias de Auto Scaling.	26 de enero de 2012

Característica	Description (Descripción)	Fecha de lanzamiento de la nueva versión
Compatibilidad con Amazon SNS	Utilice Amazon SNS; para recibir notificaciones cuando Amazon EC2 Auto Scaling lance o termine instancias EC2. Para obtener más información, consulte Recibir notificaciones de SNS cuando se escala el grupo de escalado automático. Amazon EC2 Auto Scaling también agrega las siguientes características: • La posibilidad de configurar actividades de escalado recurrentes con sintaxis cron. Para obtener más información, consulte la operación `PutScheduledUpdateGroupAction` de la API. • Un nuevo ajuste de configuración que le permite escalar sin añadir la instancia lanzada al balanceador de carga ()LoadBalancer. Para obtener más información, consulte el tipo de datos `ProcessType` de la API. • La marca `ForceDelete` de la operación `DeleteAutoScalingGroup` que indica a Amazon EC2 Auto Scaling que elimine el grupo de escalado automático con las instancias que tiene asociadas sin tener que esperar a que primero se terminen las instancias. Para obtener más información, consulte la operación `DeleteAutoScalingGroup` de la API.	20 de julio de 2011
Acciones de escalado programadas	Se ha agregado compatibilidad con acciones de escalado programadas. Para obtener más información, consulte Escalado programado para Amazon EC2 Auto Scaling.	2 de diciembre de 2010
Compatible con Amazon VPC	Se ha agregado compatibilidad con Amazon VPC. Para obtener más información, consulte Lanzar instancias de Auto Scaling en una VPC.	2 de diciembre de 2010

Característica	Description (Descripción)	Fecha de lanzamiento de la nueva versión
Compatibilidad con clústeres de HPC	Se ha agregado compatibilidad con clústeres de informáti ca de alto rendimiento (HPC).	2 de diciembre de 2010
Compatibilidad con comprobac iones de estado	Se agregó compatibilidad con el uso de comprobaciones de estado de Elastic Load Balancing con las instancia s Auto EC2 de Amazon Scaling-managed EC2. Para obtener más información, consulte Comprobaciones de estado para instancias en un grupo de escalado automático.	2 de diciembre de 2010
Support for CloudWatch alarm	Se ha eliminado el antiguo mecanismo de activación y se ha rediseñado Amazon EC2 Auto Scaling para utilizar CloudWatch la función de alarma. Para obtener más información, consulte Escalado dinámico para Amazon EC2 Auto Scaling.	2 de diciembre de 2010
Suspender y reanudar el escalado	Se ha agregado compatibilidad para suspender y reanudar procesos de escalado.	2 de diciembre de 2010
Compatibilidad con IAM	Se ha agregado compatibilidad con IAM. Para más información, consulte Control del acceso a los recursos de Amazon EC2 Auto Scaling.	2 de diciembre de 2010

Las traducciones son generadas a través de traducción automática. En caso de conflicto entre la traducción y la version original de inglés, prevalecerá la version en inglés.